Development Informatics

Working Paper Series

The Development Informatics working paper series discusses the broad issues surrounding information, knowledge, information systems, and information and communication technologies in the process of socio-economic development

Paper No. 9

Synching or Sinking: *Trajectories and Strategies in Global Software Outsourcing Relationships*

RICHARD HEEKS, S. KRISHNA, BRIAN NICHOLSON & SUNDEEP SAHAY

July 2000

Published by: **Institute for Development Policy and Management**
University of Manchester, Precinct Centre, Manchester, M13 9GH, UK
Tel: +44-161-275-2800/2804 Fax: +44-161-273-8829
Email: idpm@man.ac.uk Web: http://www.man.ac.uk/idpm

View/Download from: http://www.man.ac.uk/idpm/idpm_dp.htm#devinf_wp

Educators' Guide from: http://www.man.ac.uk/idpm/educdi.htm

Table of Contents

Synching or Sinking: *Trajectories and Strategies in Global Software Outsourcing Relationships*

Richard Heeks, IDPM, Manchester University*
S. Krishna, Indian Institute of Management, Bangalore
Brian Nicholson, School of Accounting and Finance, Manchester University
Sundeep Sahay, Department of Informatics, Oslo University
2000

Abstract

To increase the value and reduce the costs and risks of global software outsourcing, clients and developers are 'synching': building congruence along six identified 'COCPIT' dimensions. Case studies from a longitudinal research project identify practical synching strategies used by North American and European clients in their dealings with Indian sub-contractors. However, the cases also identify the limits to synching across physical and cultural distance. These limits – and the shocks to synching that occur over time – must be recognised and addressed if global software outsourcing is to succeed.

* Contact author: richard.heeks@man.ac.uk
An amended version of this paper has been submitted to *IEEE Software*

A. Global Software Outsourcing

Global software outsourcing (GSO) is the outsourcing of software development to sub-contractors outside the client organisation's home country. The first flickering was seen in the 1970s and 1980s. However, it was during the 1990s that GSO really took off. Worldwide, GSO is expanding rapidly, now involving sub-contractor countries as diverse as Ireland, China, Russia and Chile. However, India remains the unquestioned leader, registering average annual growth of more than 40% over the last decade and developing some US$3.3bn-worth of software for foreign clients in fiscal year 1999/2000[1]. Indian firms now develop software for nearly one-third of the Fortune 500[2].

Both the literature on outsourcing and its smaller cousin, the literature on global software outsourcing, have tended to focus on three key, overlapping questions:

- Why outsource: the costs and benefits of outsourcing.
- What to outsource: the type of product or service to outsource.
- How to outsource: the mechanics of successful outsourcing.

Answering the first, the key benefits/drivers underlying the specifically global fraction of software outsourcing are: cost savings, access to new labour pools, and access to new IT markets[3]. India's 'most favoured nation' status can be explained in these terms. Software wages are around one-fifth those in the US; it hosts more than one hundred thousand English-speaking software professionals; and its increasingly-liberalised local IT market spends almost US$2bn annually[1,2].

IS outsourcing more generally has undergone a change of emphasis. Outsourcing was initially seen as somewhat out of the ordinary, with an emphasis on steps necessary in order to minimise the dangers of outsourcing. Steps included: careful selection of sub-contractors; a strong understanding of processes to be transferred; a comprehensive and detailed contract; and continuous monitoring of sub-contractor performance[4].

Subsequently, IS outsourcing has been 'normalised': come to be seen as a normal part of a company's portfolio of management procedures[5]. The metaphor of outsourcing has changed from wild beast to be tamed, to beast of burden to be harnessed. Sub-contractors move from arms-length hazards to potential partners. Whilst the steps outlined above remain important, an additional aspect has therefore been added: the nature of the client—developer relationship. Successful management of that relationship by both partners shows the potential for long-term, co-operative benefits[6].

Global software outsourcing, too, has initially seen the metaphor of 'wild beast' adopted. What to/how to advice for potential clients has counselled starting small, starting at home, and starting with programmers[7]. Many client organisations have followed this advice, putting a toe into the GSO waters through small-scale body shopping: for example, having Indian sub-contractor staff come over to the client site to complete a minor, non-critical piece of coding/conversion work.

Whilst minimising risk, however, this also minimises benefits. In the US, for example, onsite costs of India-related GSO undercut those for hiring US staff by only some 10-20%. In comparison, sending development work offshore to India will typically undercut by some 50%. Large projects offer a greater potential for savings than small

ones. Likewise, the cost savings for hiring Indian analysts or project managers are typically some US$1000-2000 per month greater than those for hiring programmers[8].

Clients have therefore been keen to normalise GSO and to move up the value chain, in order to increase the benefits of outsourcing. These benefits include not merely the greater savings just noted, but also improved access to local labour and to the local IT market. Sub-contractors, too, are happy with the greater income, greater control, and greater learning to be derived from such moves.

However, moving up the value chain in this way brings additional costs and additional risks. GSO of any type imposes communication and coordination costs that local outsourcing does not have[9]. Such costs are much higher for higher-value GSO and so, too, are the risks of project failure. Clients and developers are therefore seeking routes through the cost/risk minefield to the benefits that normalised, higher-value GSO promises.

Summary of Research

The research reported here set out to investigate these routes. What is it, for example, that differentiates those who do from those who do not successfully make the move up the value chain? Drawing from the more general IS outsourcing literature, it was clear that the nature of client—developer relationships would form a key focus in answering that question.

For the reasons given above, India was selected as the location for the developer half of the relationships to be studied. Study of the dynamics of GSO relationships can only be conducted by longitudinal, qualitative, case study research. Eight GSO relationships were selected, with clients based in North America, Europe and East Asia: the major locations for software outsourcing to India. Investigation of the case studies was initiated in 1996 and still continues. To date, the authors have conducted over 300 interviews with managers and software developers in India and the various client countries.

The case studies included relationships deemed successful and those deemed unsuccessful by the parties involved. They encompassed a range of client—developer proximity from contractual relations through joint ventures to full subsidiaries. They also encompassed a range of GSO 'generations', from body shopping to part onshore/part offshore services to fully offshore turnkey contracts involving direct virtual contact between Indian developers and client customers.

B. Synching or Sinking

In seeking to understand success strategies in GSO relationships, we were struck by contradictions arising from some prescriptive recommendations. Techniques that worked well for one relationship could be a cause for friction and failure in another.

For one client, 'Gowing', the perceived deference and compliance of Indian developers was a key element in the success of the client—developer relationship. Yet, for

another client, 'Sierra' (see below), it was a major problem that contributed to the closure of their Indian operations. Most North American clients found outsourcing of highly structured work to be effective, particularly in reduction of transaction costs. But in the Japanese client contracts we studied, outsourcing less structured, more creative tasks had helped build meaningful interactions and relationships.

The analysis of field data therefore had to be informed by a contingent perspective. Classic ideas on contingency and organisation relate to the match or mismatch between organisations and their environment[10]. Such ideas are familiar from the business alliance literature[11]. They also seemed to help explain the dynamics of the GSO relationships under investigation, considering particularly the match or mismatch between developers and their 'client environment'. Successful relationships were those in which a high degree of congruence was achieved between developer and client: we called this 'synching'. Unsuccessful relationships were those in which a low degree of congruence was achieved between developer and client: we called this 'sinking'.

Congruence may exist in relation to any number of contextual dimensions. We developed a dimensional framework on the basis of our initial research and other studies of GSO and of user—developer relationships[12,13]. Using this, synching was more precisely defined as the minimisation of gaps between client and sub-contractor along the six 'COCPIT' dimensions: **C**oordination/control systems, **O**bjectives and values, **C**apabilities, **P**rocesses, **I**nformation, and **T**echnology.

Theoretical examples of complete congruence can be given for each dimension:

- *Coordination/control systems*: client and developer use the same management coordination and control systems; for example, having the same systems for staff monitoring and appraisal.
- *Objectives and values*: client and developer share the same objectives for their relationship and bring the same values to that relationship; for example, having the same organisational culture.
- *Capabilities*: this dimension is slightly different. One purpose of outsourcing is for the developer to provide human capabilities the client lacks. Congruence here therefore means that the developer's capabilities profile matches the requirements of the client; for example that, if the client needs ten Java programmers, the developer provides them.
- *Processes*: client and developer use the same work processes; for example, using the same software development methodology.
- *Information*: client and developer have access to the same information; for example, information relating to project requirements and timescales.
- *Technology*: client and developer use the same technology; for example, the same software and hardware platforms for development work.

Armed with this framework, we set out to discover the practical realities of synching.

C. Synching in Practice

Overview

Those who fail to synch, sink. Their projects run into problems, including outright failures. They fail to move up the value chain. Their relationships disintegrate rather than normalise.

For example, 'Pradsoft', a leading Indian software house, entered into a software development relationship with 'Ameriten', a large US client. Although there was some sharing of information and overlaps in technology platforms, significant congruence did not develop. The gap arose particularly because of different objectives. Ameriten saw global software outsourcing as a means to cut costs at all costs. Pradsoft, on the other hand, saw GSO as a partnership arrangement that should incorporate capacity-building and knowledge/technology transfer. Because Ameriten disagreed, transfers did not take place, and the values, processes and management systems of the two parties remained significantly different. Capabilities, too, did not develop as intended. After some initial contracts and in less than two years, the relationship was terminated by mutual disagreement, having failed to create synergy and, hence, having failed to move up the value chain.

But what of those who tried to bring the COCPIT dimensions into synch?

Complete congruence was not found in any of the GSO relationships studied. In all the more successful ones, though, synching had taken place to a significant degree along most of the COCPIT dimensions. In general terms, it was the more congruent relationships that delivered more projects closer to time and budget. They also built the pre-conditions for higher-value outsourcing. In particular, congruence had fostered trust between client and developer. It was this trust which had enabled the relationship to be characterised by progression to larger, more highly-skilled projects with more offshore components.

Nevertheless, synching in practice ran into barriers and problems, particularly around certain issues and certain COCPIT dimensions, as described in the following case studies. Client organisations can adopt a variety of specific techniques to encourage synching. In the cases presented here, these techniques sat within an overall strategic framework: the creation of an 'overseas development centre' (ODC) in India. The cases describe the way in which synching was – and was not – achieved through this strategy.

'Global' and 'Shiva'

Global is a large North American multinational telecommunications company which took a strategic decision in the late 1980s to become involved in global software outsourcing to India[14]. It began to build relationships with a number of developers, prime amongst which was Shiva, one of India's leading software exporters. Involving just ten Indian staff in 1991, the relationship grew to involve nearly 400 by the late 1990s, mainly based in Mumbai. The relationship deepened to the extent that Shiva worked on some of Global's core technology developments, and was designated a full

'partner lab'. This included transfer of ownership of some software products from Global to Shiva, and direct contact between Shiva and some of Global's own customers.

For this depth of relationship to be achieved, 'the principles and the overall process of attaining business goals have to be congruent', as one of Global's general managers stated. So how – and to what extent – has Shiva been brought into synch with Global?

Global sees itself as having worked to achieve congruence on all the COCPIT dimensions. It has succeeded quite well on what can be termed the 'harder dimensions' such as processes and technology. Network links have been steadily upgraded until Shiva has the same high-speed links and the same access to Global's corporate WAN as any of Global's own departments. Global's own software development environment has also been replicated in India. Global has set aside a dedicated group to ensure that matching technical resources are available in Shiva.

For each project, there has been a detailed process of project definition and specification development. This ensured that project methodologies, scope, schedule and deliverables are unambiguously defined and understood by both parties, helping synch information and processes. Contract negotiations have similarly been used as a mechanism for building shared understanding, even attempting to address some of the softer components of synching, such as objectives and values.

At both senior and project level, Global has made extensive use not only of the communications facilities available, but also of physical movement in order to provide for face-to-face meetings. Such meetings were found to be more effective at synching values and informal information, in a way that ICT-mediated communication could not. Global has also ensured that traffic is two-way, with Global staff visiting Shiva's offices in India as well as having Indian staff come to North America.

Global instituted a comprehensive programme of training for Shiva staff to bring capabilities into line with its requirements. This also addressed the process dimension, for example enabling Indian staff to follow Global's software development methodologies. It covered objectives and values too by endeavouring to transmit Global's corporate culture and value system. Periods of work for Shiva staff in Global's North American office attempted to do the same thing.

Finally, Global also tried to address the coordination/control systems dimension. It supported the introduction of North American HR management systems. Despite barriers, Shiva introduced performance appraisal and career management systems like those operating at Global headquarters.

These techniques built a significant degree of congruence, especially on the capabilities, processes, information and technology dimensions. Shiva's operation was run quite like those in Global's home country. The result was average annual growth of more than 15% in Shiva—Global business. By the late 1990s, Global was a major Shiva client, accounting for some US$8million-worth of outsourcing contracts per year.

However, Shiva remains an Indian organisation based in India, and limits to synching emerged from our fieldwork. Perhaps not surprisingly, the limits centre around the

soft issues of objectives and values since these encompass deep socio-cultural differences that not easily erased. They have had a knock-on effect on coordination/control systems. Two examples can be cited:

Programmer attrition rates. Indian programming is characterised by strong demand—supply gaps, sourcing from elite educational institutions, vast salary differentials between employment opportunities, and an obsessive search for new, globally-marketable skills. The result is that Shiva understands and, is partly imbued with, a culture of both continuous project-hopping and job-hopping.

This software project culture did not match Global's telecommunications product culture. In the latter, 15-20 year tenures were common and Global managers brought these norms and expectations to their relationship with Shiva. They saw attrition as a major problem to be prevented by what was described as a 'Berlin Wall approach' to control. Shiva managers, however, took attrition more in their stride as something to be managed rather than stopped. As one senior Shiva manager commented:

> The "biological clock" of Indian software programmers is 15 weeks, and they then move to a different project. Global has an obsession with attrition while here we try to work with attrition as a fact of life.

The culture of management systems. Synching coordination/control systems involved the attempted displacement of Shiva's systems by Global's own. This did not run smoothly because of the cultural underpinnings of those systems. Shiva's systems were deep-rooted and based on a relatively personalised and subjective management culture that had a long tradition. Global's systems were drawn from a culture of objectivity and accountability. Forcing one set of values onto the other was hard, and Shiva's values proved quite resilient. It took enormous efforts before the Shiva project leader would produce a standardised monthly progress report, and Shiva staff refused to participate in Global's employee satisfaction survey. Shiva therefore came to be seen as an 'outlier' in the 'Global family' because its coordination/control systems and its objectives and values could not be pushed fully into synch.

Congruence of this client—developer relationship has also remained subject to factors in the external environment. During the 1990s, Shiva was made and made itself significantly congruent with Global and its expectations and requirements. In the late 1990s, e-business models arrived, and Global's IS strategy took a sudden left turn into Web-based models and applications. Shiva was – and was seen by Global to be – in synch with legacy-based models and applications, not with the new world of e-business. Synching on every one of the COCPIT dimensions, especially capabilities and processes, began to spring apart. At the time of writing, the relationship was struggling to get back into synch, having been thrown out by this environmental change.

'Sierra'

Sierra is a small but rapidly expanding UK software house, specialising in high-tech bespoke projects[15]. By 2000, it had a turnover of roughly US$12million and employed 80 staff worldwide. Sierra began outsourcing development work to India in 1996 using a body shopping model. Congruence was poor: capabilities were not up to expectations and, more importantly, Indian staff lay outside Sierra's HR systems and company culture, creating factional divisions between Indian and UK staff.

To bring the developers more in synch, Sierra set up an overseas development centre in Bangalore in 1998. The centre was created on the basis of a rather hastily-conducted business plan but with a high level of optimism and expectation about what could be achieved. The main intention was to produce "a little bit of Sierra in India".

Firm synching strategies were put into place. The ODC was incorporated as a full subsidiary of Sierra, run by a UK manager. Sierra's home processes and coordination/control systems were introduced. High-speed communications links with video-conferencing functionality were installed, synching the technology dimension. Information was shared as it was in the UK with direct access to the corporate intranet. A 'youthful and unstructured' work environment, like that of UK headquarters, was envisaged.

Blinded by its enthusiasm, Sierra thought it saw congruence on all the COCPIT dimensions, and shot the relationship up the value chain. Whole projects, including 'virtual liaison' with Sierra customers, were outsourced to the Indian team, envisaging that distance in all its connotations – geographical, cultural, linguistic – would be invisible.

Some projects were deemed successful; especially those that involved members of the Indian development team travelling to the customer site in Europe for requirements capture. However, as this example illustrates, distance could not be made invisible and limits to synching emerged. Some could be addressed: customers found some accents hard to understand, creating a barrier to information synching. Indian staff were sensitised and sent to evening classes. One interviewee, with a notable 'British English' accent, stated this was for work-time use; like others, he reverted to his 'Indian English' accent at home.

Other differences, though, were more persistent. As might be expected, the video-conferencing link (when not blanked out by bad weather) could not substitute for face-to-face interaction. It failed to transmit the informal information that personal contact provides, again creating a barrier to information synching.

However, as with Global—Shiva, most of the persistent differences clustered along the 'objectives and values' dimension: the dimension, arguably, about which Sierra assumed most and did least to achieve congruence. These differences, in turn, undermined other COCPIT dimensions. A number of examples of cultural dissonance emerged from the case study:

Drinking rituals. These feature highly in Sierra's UK working culture. The ability to talk with senior staff after work over a drink is even highlighted in the company's

UK recruitment literature. Alcohol-fuelled activities underpin the informal, team-based work processes of the UK offices. They provide: the venting channels for work and inter-personal stresses; the forum for discussion of problems; and the initiation mechanisms for new recruits. In India, life was different as the ODC manager noted:

> The London office is much freer. A typical example from the UK is when someone joins us we open a bottle of champagne. People casually mill around, say 'Hi' and move on. Here I am expected to make a speech. 20% drink, 80% don't. In the UK everyone drinks.

Authority and creativity. In many ways the issue of alcohol is a window onto deeper issues of authority and hierarchy. In Sierra UK, authority comes more from technical knowledge rather than position. Processes of 'creative discussion' were legitimised, meetings were open and confrontational and, during problem-solving discussions, junior staff could and did openly contradict more senior staff. In Sierra's Indian office, despite the congruence achieved on some dimensions, a different attitude to authority and to confrontation prevailed, as the ODC manager described:

> I am not allowed to enter into their [*Indian staff*] comfort zone. I am the manager, they will not let me talk to them and they will always agree with me. It is really annoying, sometimes I tell them something that is wrong. I want to hear "I don't agree with you". Here they are all interested in the 'tag' [*formal designation of rank*].
>
> In meetings, the staff will often stay quiet and then I will get a lengthy email maybe an hour later when the meeting is over and I have to start the thing all over again and I think , well why not bring all that up in the meeting? It ends up taking twice the time. They will not confront me in meetings, the spark is missing. They lack ability to ask questions, lack enthusiasm … a thirst for knowledge.

Of course there is no right and wrong here – the quotes told us as much about the UK manager as they did about Indian staff. But what they did highlight was a gap of values and processes that was hard to bridge.

Project leakage. Sierra managers were concerned about 'project leakage': slippage in the amount of time dedicated to a project. In the UK, leakage was typically 2-3%; in India, it was typically 25-30%. The problem arose over definitions of leakage. From the UK perspective, leakage was measured not in terms of deadlines, but in terms of how much time was spent on a task. From the Indian staff's perspective, it was deadlines that mattered and UK values were insensitive to Indian conditions. Components of those conditions included:

- Bangalore's strained, overcrowded infrastructure where travel delays and power cuts were common,
- a relatively large number of public holidays, and
- a view of home/work balance that incorporated the execution of family responsibilities (paying bills, taking family members to the doctor, etc.) during working hours.

Indian staff compensated for these by working additional hours to ensure that deadlines were met. But in the UK scheme of things this was all treated as leakage.

Differences in cultural values therefore affected other dimensions that are underpinned by culture. To some degree, coordination/control systems, capabilities and processes in the Bangalore office were all constrained from synching with the realities or expectations at UK headquarters.

These limits to synching within an ODC framework were combined in the late 1990s with an external 'shock to synching' similar to that suffered by Global—Shiva. Sierra moved aggressively into the new e-commerce market where contracts had short lifecycles and a strong requirement to understand a lot of specific customer needs.

The limits and shocks threw the client—developer relationship severely out of gear on several dimensions and, in 1999, Sierra's UK managers closed down their Bangalore operation. Having failed to 'culturally flood' developers in India, they decided to force synching by repatriating development activities, along with Indian staff, to Britain. This was an attempt to disembed those staff from their cultural infrastructure and to enforce congruence on the foundational but stubborn objectives and values dimension.

Developer Synching Strategies

The focus of the case studies reported here has been on synching strategies adopted by client organisations, since they were found to be 'in the driving seat' of GSO relationships. However, developers also made use of synching strategies. In many cases, these were generic strategies, aiming to bring the Indian firm's coordination/control systems, objectives and values, capabilities, processes, information and technology more into line with those found in a typical major Western client organisation.

Quality certification schemes like ISO9000 represent a generic synching method. Some interviewee firms had undertaken certification within the context of a specific GSO relationship, but more had used it as a 'pre-synching' strategy. The intention in the latter case is that certification gives developers a template for moulding themselves at least part-way into synch with potential clients on at least some of the synching dimensions: particularly dealing with coordination/control systems and processes and, to a lesser extent, with objectives and values, capabilities and information. Certification is a signal to those clients that says something like "Even though we come from a very different country, we're not actually that different: we are significantly in synch with you and with what you require from a software developer".

So powerful is certification as a synching tool/strategy for GSO relationships that it has spread like wildfire through the Indian software industry. By April 2000, more than 210 Indian software firms had acquired ISO9000 or CMM quality certification, with a further 70 expected to certify by the end of 2000. India will soon have more quality-certified software firms than any other nation. It already has 10 of the 18 companies worldwide that are certified to the highest quality level[15].

Developers also use the mirror image of the client's overseas development centre. They create an overseas office outside India that can be used to provide feedback. This feedback is then used by the developers to bring their technology, processes, systems, etc. more into synch with those of potential clients. The value of this as a synching tool/strategy is reflected in the fact that, by April 2000, 212 Indian software firms had either subsidiaries or branch offices overseas, mostly in the US[16].

Finally, developers in long-term relationships sometimes find themselves at risk of growing out of synch with their clients. 'Davari' and 'SRS' were both developers for Global. A fairly congruent relationship with Global developed quite rapidly, based around relatively routine outsourcing work largely undertaken offshore. Over time, both Indian firms began to change and congruence was in danger of being lost. The developers' managerial systems, aspirations, capabilities and quality processes started to run ahead of those required by their GSO relationship.

In theory, synching could then have involved bringing the client up to the developer's standard. Yet this did not happen. Whether initiated by client, developer or both, synching invariably meant making the developer like the client rather than vice versa. Client expectations did not encompass the idea that their Indian developers might have something to offer in the way of good, let alone best practice. Western clients failed to see synching as anything other than one-way traffic to a very much junior partner in the relationship.

This led some of those junior partners to diversify their talents away to a more appreciative audience. They adopted strategies of partitioning and diversification. A separate unit was set up in each Indian firm, in which COCPIT dimensions were held in synch with those of Global. The developers then sought out new and additional GSO relationships that were more in synch with their mainstream dimensions. SRS, for example, took on contract work from Japanese firms which required SRS to set up its own R&D teams to develop a set of complex algorithms for software control of telecommunications networks. This provided a good match to its newly-developed systems, objectives, capabilities and processes.

D. Conclusions and Recommendations

Table 1 summarises three of the cases presented above.

Table 1: Client—Developer Relationships, Synching and Outcomes

Relationship	Degree of Synching						Outcome
	C	O	C	P	I	T	
Pradsoft—Ameriten	x	xx	√	x	√	√	Insufficient synching leads to relationship failure
Global—Shiva (pre-shock)	√	—	√√	√	√	√√	Synching just sufficient for relationship development
Sierra UK—Sierra Bangalore (pre-shock)	—	—	√	√	—	√√	Synching sufficient for relationship survival but not development

√√: very congruent; √: fairly congruent; —: partly congruent; x: slightly congruent; xx: not congruent

From our experience of these and other cases, synching – creating congruence between client and developer along a number of dimensions – lays the foundation for higher project success rates in GSO and for higher-value GSO. Synching is a pre-requisite for moves up the 'trust curve' that leads to higher-value GSO. It is also a means of reducing the risks and costs associated with higher-value GSO. It is a key component in the 'normalisation' of global software outsourcing.

Both client and developer managers must understand what synching means in their particular context, and must look for strategies – like those described above – that bring about synching. But they must also recognise the serious practical difficulties of synching. For example, whilst it proved relatively easy to synch technology and capabilities, it proved hard to synch some aspects of information and coordination/control systems and very hard to synch objectives and values. This created limits to synching.

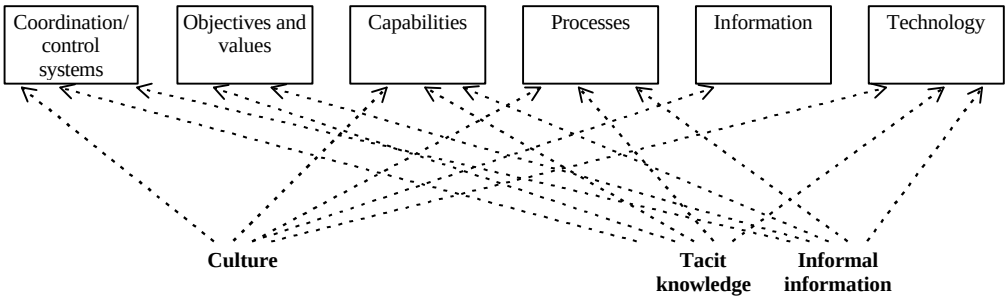
Limits to Synching

Western clients seem to fall too easily for the line of the 'ultraglobalists' who argue that, in a globalised world, distance, borders and place no longer matter. The experience of these cases suggests otherwise.

Distance does still matter, and it interferes with synching because of the difference between the formal/tangible and the informal/intangible aspects of GSO. Synching strategies in practice have been good at dealing with the former and poor at dealing with the latter. In particular, they have been poor at dealing with three overlapping issues: tacit knowledge, informal information, and culture.

In direct terms, the first two are part of the information dimension, and the latter is part of the objectives and values dimension. However, their importance derives from the way in which each indirectly affects all five COCPIT dimensions other than its own, as shown in Figure 1.

Figure 1: The Influence of Tacit Knowledge, Informal Information and Culture



Technologies, specifications, processes, methodologies, skills, objectives and management systems can be transferred from client to developer. But all these have informational components consisting of two parts: the explicit knowledge that can be laid out formally, and the tacit knowledge that cannot. The former is readily transferred; the latter is not. Global, for example, went with GSO best practice and used clear, formalised requirements specifications. Whilst necessary, these were not sufficient because they incorporated a whole set of tacit assumptions and understandings that were not transferred: about the nature of the customer, about design and programming choices, about working practices.

Clients especially must therefore pay more attention to tacit knowledge. They must recognise its existence within all the COCPIT dimensions, identify its content, and look for ways to synch it between themselves and their developers.

Ready transfer of formal procedures also runs into problems because real software development requires constant divergence from formal guidelines and requires constant improvisation. Informality and improvisation require informal information which thus remains a critical resource for global software development.

Trying to focus on well-structured, stable projects enabled some case study clients to push a lot of information exchange into the formal realm that can be handled relatively well by ICT-mediated distance. But informal information could not be handled that way. Sierra found the cost, fragility and artificiality of video-conferencing excluded informal conversations and restricted interaction to formal exchange of progress towards milestones. Email, too, only operated at an informal level once participants had physically met and built personal relationships. Travel and direct meetings will therefore remain a continuous and crucial element in GSO relationships to help more fully synch the information dimension.

Thirdly, players in this global game all still retain cultural values rooted in a particular locale. The overseas development centre is a powerful tool for synching and, thus, for raising project success rates and for moving up the GSO value chain. But it has its limits. Western systems, capabilities, processes, etc. can all be imposed. However,

some cultural 'stains' underpinning these dimensions are hard for this global tide to wash away.

Hiring foreign IT workers as client company staff for onsite operations – as Sierra has now done – may scrub those stains even harder, forcing developers into synch. But this is hardly global software outsourcing and it reaps few of the intended benefits of GSO.

Living with Limits to Synching

How can Western clients retain the benefits of GSO and yet live with the limits to synching? Some pointers were noted above. These are specific examples of a more general need to create buffering mechanisms between the distant worlds of client and developer, and bridging mechanisms that allow 'intangibles' to be exchanged between those worlds. Other examples include:

Management of expectations. Sierra's overseas development centre foundered, in part, because UK expectations were out of synch with Indian realities on several COCPIT dimensions. Those expectations – of a Bangalore where the streets are paved with programmers, where technology can destroy distance, where Indian staff become British clones given the right environment – were built on too much media hype and too little cold, hard analysis. As the overly positive expectations crashed down to reality, they left in their wake a disillusionment with GSO and a poor image of India.

A key, then, to successful GSO is a realistic expectation of what can be achieved given the level of attrition, the limitations of the technological infrastructure, the cultural differences, and so forth. Global and Shiva built up a good in-house understanding of what can and cannot be achieved over a long period with a slow-growing expansion and trust curve. This was more successful than Sierra's 'too much, too soon' approach.

Using straddlers. Straddlers are those who have a foot in the world of the client and a foot in the world of the developer. Global and Shiva had many of these. Global, for example, has used some of its India-born managers in GSO relationships. These staff have proven adept at understanding what can and cannot be brought into synch. They manage the dimensional gaps that remain, often acting as a buffer between the Indian developers and senior client managers. Shiva, too, despite attrition rates, has staff at all levels with significant experience of working in North America who are better able to see things from the client's perspective. Those straddlers have also acted as conduits for transfer of tacit knowledge and informal information. Sierra, by contrast – perhaps because it is a smaller and more recent entrant – had no effective straddlers in either client or developer groups.

Building bridging relationships. Straddlers bridge gaps within one individual. An alternative is to bridge gaps by building strong one-to-one relationships between individual clients and developers. Global and Shiva have been active in this, facilitating meetings and other informal fora for contact in which such relationships can develop. They have even proactively identified pairs of individuals to bring

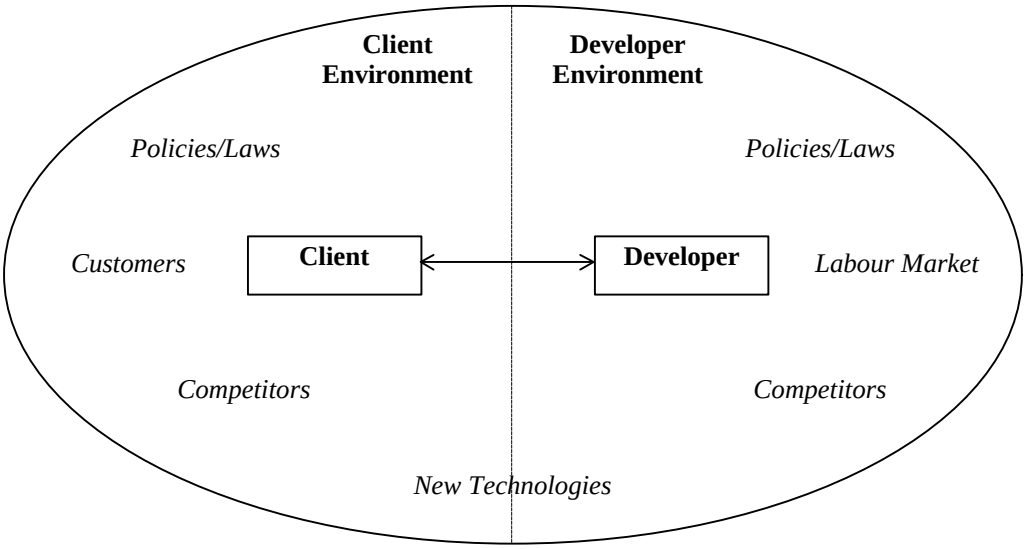
together. These relationships have been a valuable means by which to cope with continuing client/developer mismatch. They create a channel for translating between client and developer, and for passing informal information and tacit knowledge. They also create the 'synch' of mutual trust and understanding that can help overcome other dimensional differences.

In addition to their synching strategies, clients and developers must therefore also identify the buffering and bridging mechanisms that will help them deal with the limits to synching that still exist within global software outsourcing for some key COCPIT dimensions.

Understanding the Synching Environment

Both main cases exposed the danger of external 'shocks to synching', and they are a reminder that the client—developer relationship does not sit in vacuum. Instead, it sits within an environment of which some components are illustrated in Figure 2.

Figure 2: The GSO Relationship Environment



As well as synching with each other, clients and developers also seek to synch with other components of their environment. This creates pressures and tensions for the client—developer relationship.

These may be acute: the negative impact of e-business developments. Or they may be chronic: the problems of developers in meeting the demands of staff in the local labour market. These demands related to wages, development of state-of-the-art skills, travel, and length of project assignments. As Global and Shiva discovered, such demands may run counter to what the client provides, leading to attrition levels way above what the client expects.

There is no magic solution to meeting these environmental pressures, especially when they are unexpected. Explicit discussion of the pressures will be one step; diversification of both parties into other relationships will be another.

Managers must recognise that there is a seventh dimension to synching: time. Congruence must be actively sustained because, once created, it cannot ensure a permanently 'synched' relationship in a context of continuous environmental change. Clients and developers must therefore continuously re-examine their relationship and proactively move to address emerging mismatches.

References

1. R.B. Heeks, *India's Software Industry*, IDPM, University of Manchester, UK, 2000. www.man.ac.uk/idpm/isi.htm

2. Dataquest, *The DQ Top 20: Volume 1*, Dataquest (India), New Delhi, July 15, 1999.

3. J.D. Herbsleb and R.E. Grinter, "Architectures, Coordination and Distance: Conway's Law and Beyond", *IEEE Software*, Vol. 16, No. 5, Sept./Oct. 1999, pp. 63-70.

4. M.C. Lacity and R. Hirschheim, *Information Systems Outsourcing: Myths, Metaphors and Realities*, John Wiley, Chichester, UK, 1994.

5. S. Selinger, *Normalisation of Outsourcing*, MSc Thesis, London School of Economics, 1996.

6. R. Elitzur and A. Wensley, "Game Theory as a Tool for Understanding Information Services Outsourcing", *Journal of Information Technology*, Vol. 12, 1997, pp. 45-60.

7. F. Warren McFarlan, "Issues in Global Outsourcing", *Global Information Technology and Systems Management*, P.C. Palvia et al., eds, Ivy League Publishing, Nashua, NH, 1996.

8. R.B. Heeks, *India's Software Industry*, Sage Publications, New Delhi, 1996.

9. U. Apte, "Global Outsourcing of Information Systems and Processing Services", *The Information Society*, Vol. 7, 1997, pp. 287-303.

10. P. Lawrence and J. Lorsch, *Organization and Environment*, Harvard Press, Harvard, MA, 1967.

11. B. Kogut, "Joint Ventures: Theoretical and Empirical Perspectives", *Strategic Management Journal*, Vol. 9, 1988, pp. 319-332.

12. R.B. Heeks, "Why Information Systems Succeed or Fail: Conception-Reality Gaps", *Proc. BIT '98 'Business Information Management' Conf.*, Manchester Metropolitan University, UK, 1998.

13. E. Carmel, *Global Software Teams*, Prentice Hall, Upper Saddle River, NJ, 1999.

14. S. Sahay and S. Krishna, *Understanding Global Software Outsourcing Arrangements: A Dialectical Perspective*, Center for Software Management, Indian Institute of Management, Bangalore, 1999.

15. B. Nicholson, S. Sahay and S. Krishna, "Work Practices and Local Improvisations Within Global Software Teams: A Case of a UK Subsidiary in India", *Proc. IFIP WG9.4 'Information Flows, Local Improvisations and Work Practices' Conf.*, IFIP WG9.4, Cape Town, South Africa, 2000.

16. N. Kumar, *New Technology Based Small Service Enterprises and Employment*, International Centre for Development Research and Cooperation, New Delhi, 2000.