This is the post-peer reviewed final draft version of the following article: Heeks, R., Krishna, S., Nicholson, B. & Sahay, S. “Synching or sinking: global software outsourcing relationships”, *IEEE Software*, March/April 2001, 54-61, which has been published in final form at DOI: 10.1109/52.914744

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

Submission for IEEE Software Special Issue on 'Global Software Development'

Richard Heeks, Manchester University
S. Krishna, Indian Institute of Management, Bangalore
Brian Nicholson, Manchester University
Sundeep Sahay, Oslo University

Abstract

Clients and software developers need to move their global outsourcing relationships up the value chain in order to reap greater benefits. Yet such moves bring costs and risks. We therefore set out to investigate the strategies that differentiate successful and unsuccessful value chain moves. Case studies from longitudinal research identified 'synching' as a key success strategy: building client-developer congruence along six identified 'COCPIT' dimensions. Those who fail to synch, sink: they freeze on the value chain and their relationships founder. However, synching can be problematic because distance still matters in our supposedly 'borderless' world. Distance particularly constrains the synching of tacit knowledge, informal information and cultural values. Clients must recognise this – and the external shocks to synching that occur over time – and adopt appropriate 'buffering' tactics.

Global software outsourcing (GSO) is the outsourcing of software development to sub-contractors outside the client organisation's home country. India is the leading GSO sub-contractor, registering average annual growth of more than 40% over the last decade and developing nearly US$4bn-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].

Advice for potential GSO clients counsels starting small, starting at home, and starting with programmers[3]. 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, this also minimises benefits. In the US, 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[4].

Clients have therefore been keen to move up the value chain, in order to increase the benefits of outsourcing. These include not merely the greater savings just noted, but also improved access to local labour and to the local IT market.

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[5]. It also imposes risks of total project failure (software not delivered/unworkable) or partial project failure (cost/time overruns, failure to meet other client requirements). Costs are much higher for higher-value GSO and so, too, are risks of project failure. Clients and developers are therefore seeking routes through the cost/risk minefield to the benefits that higher-value GSO promises.

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.

Eight GSO relationships were investigated through longitudinal, qualitative case study research initiated in 1996. For the reasons given above, India was selected as the location for the developer half of the relationships to be studied. Relationship clients were based in North America, Europe and East Asia: the major locations for software outsourcing to India. In the discussion that follows, fictitious names are used to preserve anonymity.

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', 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[6]. Such ideas are familiar from the business alliance literature[7]. 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[8,9]. 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.

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.

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 or, hence, 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 a progression to larger, more highly-skilled projects with more offshore components.

Nevertheless, synching in practice run into barriers and problems, particularly around certain issues and certain COCPIT dimensions, as described in the following case studies.

'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[10]. 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.

Global sees itself as having worked to achieve congruence on all 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 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 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 operations were 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 are not easily erased. They have had a knock-on effect on coordination/control 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, Global's IS strategy took a sudden left turn into Internet-based models and applications. Shiva was – and was seen by Global to be – in synch with legacy-based models and applications, not with this new world. 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[11]. 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 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 for requirements capture. However, as this example illustrates, distance could not be made invisible and limits to synching emerged. 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, creating a barrier to information synching.

As with Global—Shiva, though, 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. Two examples of cultural dissonance can be given:

Authority. 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*].

Of course there is no right and wrong here. Instead, there is 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 work/life 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.

CONCLUSIONS

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—Bangalore (pre-shock)	—	—	√	√	—	√√	Synching sufficient for relationship survival but not development

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

There are many causes for success and failure of GSO. However, from our experience of the cases described above and others, 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.

Both client and developer managers must understand what synching means in their particular context. They must recognise which dimensions are most important in that 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.

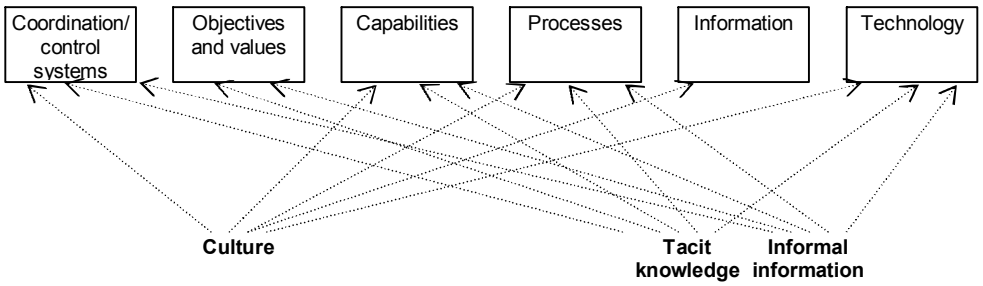
Limits to Synching

Western clients seem to fall too easily for the argument 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; particularly 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



Tacit knowledge: 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 COCPIT dimensions, identify its content, and look for ways to synch it between themselves and their developers. Techniques may include those described below: use of physical meetings, straddlers, and bridging relationships.

Informal information: 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.

Culture: 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 processes, systems, capabilities, etc. can all be imposed. However, some cultural 'stains' underpinning these dimensions are hard for this global tide to wash away. As described next, clients must learn to live with this.

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.

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 high 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.

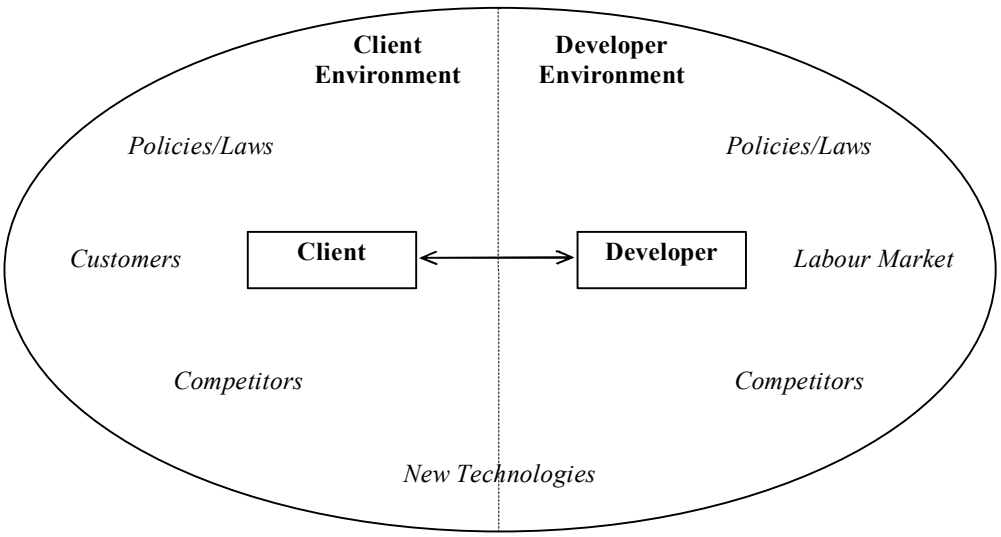
Building bridging relationships. Straddlers bridge gaps within one individual. An alternative is to bridge gaps by building strong one-to-one relationships between individual clients and developers. Global and Shiva have been active in this, facilitating meetings and other informal fora for contact in 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 demands of staff in the local labour market. These demands relate 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. NASSCOM, *Annual Review of the Indian Software Industry*, New Delhi, 2000.

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

3. 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.

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

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

6. P.R. Lawrence and J.W. Lorsch, *Organization and Environment*, Harvard Press, Harvard, MA, 1967.

7. B. Kogut, "Joint Ventures", *Strategic Management Journal*, Vol. 9, 1988, pp. 319-332.

8. R.B. Heeks, "Why Information Systems Succeed or Fail", *Proc. BIT '98 'Business Information Management' Conf.*, Manchester Metropolitan University, UK, 1998 (CD-ROM).

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

10. S. Sahay and S. Krishna, *Understanding Global Software Outsourcing Arrangements*, Technical Report, Center for Software Management, Indian Institute of Management, Bangalore, 1999.

11. B. Nicholson, S. Sahay and S. Krishna, *Work Practices and Local Improvisations Within Global Software Teams*, Working Paper, School of Accounting and Finance, University of Manchester, 2000.

About the Authors

Richard Heeks is a senior lecturer in information systems at the Institute for Development Policy and Management, University of Manchester. His research interests focus on the role of IT in governance and in international development. His PhD researched the Indian software industry.

S. Krishna is a professor at the Indian Institute of Management, Bangalore. He holds a PhD in software engineering and chairs IIMB's software enterprise management programme, focusing on research and management education in partnership with the local software industry. His research interests concern global software work arrangements.

Brian Nicholson is a lecturer in information systems at the School of Accounting and Finance, University of Manchester. His research interests focus on understanding the complexities of software development between the UK and India, which was also the topic of his PhD.

Sundeep Sahay is an associate professor at the Department of Informatics, University of Oslo. His research interests concern globalisation, IT and work arrangements. Over the past four years he has been involved in an extensive research programme analysing processes of global software development using distributed teams.

Contact Richard Heeks at IDPM, University of Manchester, Precinct Centre, Manchester, M13 9GH, UK; richard.heeks@man.ac.uk

Keywords
outsourcing, global software outsourcing, India, best practice, models, congruence