Chapter 6
Trials and Tribulations of the Global Software Engineering Process: Evolving with Your Organisation

Oisín Cawley

Abstract This chapter will provide the reader with a firsthand account of the trials and tribulations of working in and managing a Global Software Engineering (GSE) function. By describing the move from a distributed collection of self-sufficient manufacturing plants with locally managed software engineering resources, to a GSE function as a shared service, the focus will be on how the management of that group had to fundamentally change in order to satisfy the complex projects and customer base which resulted. In parallel it will discuss the effect of regulation on the software engineering management process. Tracing the introduction of financial systems regulations, it will discuss the issues this brought to the GSE process and how they were successfully overcomed. These topics will be augmented by research that the author has carried out into regulated software development.

6.1 Introduction

The term Global Software Engineering (GSE) is fairly well understood within both industry and academia, but the devil is in the detail. Companies or Information Technology (IT) departments do not develop efficient software engineering functions at a global level overnight. Typically, these companies will be large organisations with offices in multiple countries and/or geographical regions, and are therefore subject to the well documented effects of separation (geographical, temporal, and cultural). However, such companies often come from humble beginnings, and through various forms of growth (organically or through mergers and acquisitions), evolve into organisations which necessarily must function differently. It follows, therefore, that the business processes which have been in place at the beginning must change or evolve in tandem with the organisation's growth. The Software Engineering function is one of those key processes, and it is not immune to these changes.

O. Cawley (✉)
Department of Computing, Institute of Technology Carlow, Kilkenny Road, Carlow, Ireland
e-mail: oisin.cawley@itcarlow.ie

M. Kuhrmann et al. (eds.), *Managing Software Process Evolution*,
DOI 10.1007/978-3-319-31545-4_6

This evolution can cause a lot of organisational pain as the work load increases and diversifies. The need for additional resources and personnel usually requires a change to team structures, reporting lines and perhaps job descriptions in order to support the business on a more global scale. As competition in the market place increases, cost pressures often make companies look at outsourcing options. This alone can be a major challenge, but combined with the effect of having to adhere to newly created regulatory controls raises the bar significantly.

The author's experiences bears witness to a lot of what has been mentioned above, having spent 17 years working on software projects, large and small, in both small and multi-national companies. In addition, his research into the effects regulatory controls have on the software engineering function, contributes an expert insight on the topic.

6.2 Background and Context

Managing a team of highly skilled individuals, who are located in different countries, often in different time zones, to successfully deliver a software project on time and within budget can be a difficult task [7]. Table 6.1 from [2] provides an overview of the key software development process difficulties which can be encountered due to the effects of the different dimensions of "distance."

Often, such a GSE function exists as a virtual shared service for the entire organisation [16]. This can help to eliminate resource duplication, and maintain some semblance of development standards. It can however introduce a bottleneck within the organisation, as each business unit vies for priority on the software development queue [12]. This raises an interesting question. Then how should projects be prioritised? In a global organisation, who really controls the GSE resources?

6.2.1 Positioning the GSE Function

Before looking at the area of control in more detail, let us first draw a distinction, in broad terms, between two types of organisations who may employ such a GSE function. Group one are companies whose primary business is to sell software. They may produce the software in a number of ways, in-house, outsourced or a combination of the two. The in-house approach can also be implemented in two forms; collocated, where the software engineers sit together in the same place, or global, where they are geographically spread-out. Such companies may also have some form of support agreement that they sell with the software, but their focus is/should be on selling high quality software. The higher the quality, the lower the failure rate and consequent need for costly maintenance.

Table 6.1 An overview of the framework of issues in Global Software Engineering

	Dimension		
Process	Temporal distance	Geographical distance	Socio-cultural distance
Communication	Reduced opportunities for synchronous communications, introducing delayed feedback. Improved record of communications	Potential for closer proximity to market, and utilization of remote skilled workforces. Increased cost and logistics of holding face to face meetings	Potential for stimulating innovation and sharing best practice, but also for misunderstandings
Coordination	With appropriate division of work, coordination needs can be minimized. However, coordination costs typically increaser with distance	Increase in size and skills of labor pool can offer more flexible coordination planning. Reduced informal contact can lead to reduced trust and a lack of critical task awareness	Potential for learning and access to richer skill set. Inconsistency in work practices can impinge on effective coordination, as can reduced cooperation through misunderstandings
Control	Time zone effectiveness can be utilized for gaining efficient 24x7 working. Management of project artifacts may be subject to delays	Difficult to convey vision and strategy. Communication channels often leave an audit trail, but can be threatened at key times	Perceived threat from training low-cost 'rivals'. Different perceptions of authority/hierarchy can undermine morale. Managers must adapt to local regulations

Group two are those companies where the software they develop is merely a tool to support the company's business objectives (whatever they may be). We do not underestimate the importance of the software, however, for these companies the focus is on selling something else, and software is a supporting/enabling mechanism. Such companies may also take an in-house, outsourced or combination approach. This is an important distinction between the two groupings, as it positions the GSE function relative to the business objectives. It is also important because the software often plays a critical role in one or more business processes. For example, delivering inventory to a manufacturing line in a just-in-time fashion requires the order management, warehouse management and purchasing management departments to function in a coherent manner. Any issues encountered in the supporting systems have the potential to stop everything.

The two groupings here are sometimes referred to as Packaged versus IS (information systems) teams. Packaged teams normally produce an end product. This product is packaged up and sold commercially. IS teams are generally considered to be working internally to support corporate objectives. Carmel and Sawyer [8] state that the differences between IS and Packaged software teams include cost pressures versus time-to-market pressures, and bureaucratic versus entrepreneurial cultural milieus.

There are, of course, companies who do not fit within either of these broad groups, such as charities and other not for profit organisations. Such organisations tend to have different objectives and work ethea, for example, volunteer employees, and so some aspects of this chapter may not be so relevant in their cases.

6.2.2 The GSE Problem

Managing a GSE process brings new challenges. Let us consider, for example, the maintenance phase of a typical software development lifecycle (SDLC) of a company from group two. When a production issue occurs (let's say the order management system above loses a customer order, and so that particular order never makes it onto the production schedule), it must be acted on immediately. It becomes an "all hands on deck" situation because the business process is suffering. In these situations you need personnel who are qualified and trained to troubleshoot the issue. Once the issue is identified, for example, a corrupt index in the database, you need the appropriate personnel (often different to the people who did the trouble-shooting) to fix the issue. You may have these resources on site and under your control, and so issues can be reacted to rapidly. But things are quite different within a global setting. No longer are your resources local, perhaps not even in your geographical region, and in addition they more than likely report to a different manager. It is clear that the process to resolve issues cannot be managed the same in both organisational structures.

6.2.3 Regulation

According to [6], regulations are simply a form of social organisation: *"rules, principles, or conditions that govern procedure or behaviour"* [33]. But why do we have or need regulations? Fundamentally it is because we want to try and ensure a certain outcome. They provide a blueprint for how something should be done, and if we follow the rules we should end up with a good quality product or process. There are those, however, who argue that there is too much regulation and that not enough research has been done in assessing the adequacy of regulations in achieving their intended aims [6]. For example, regulations governing financial institutions, such as the Basel II Accord [3] and the Sarbanes–Oxley Act [1], did not prevent the global banking crises of 2007.

Software Regulation In relation to software, regulations are becoming more and more prevalent due to software becoming so pervasive in society. We have come to rely on it more and more in our daily lives [18] and consequently, when it fails or is misused, the effects can be quite devastating. To counteract this risk, various authorities have introduced regulations which aim to govern how software is developed, secured and interacts with other systems. For example, the Enron scandal 2001,

which resulted in the loss of over $11 Billion of investors' and employees stocks and pensions, was due to fraudulent financial reporting [4]. In Panama, 21 patients died from overdoses of radiation during cancer treatment as a result of software failure combined with software misuse [5]. FDA analysis of 3,140 medical device recalls between 1992 and 1998 found that 242 (over 7 %) were attributable to software [20]. Significantly, of the 23 recalls in 2007 of what the FDA classify as life-threatening, 3 of them involved faulty software [23]. Consequently, regulations have been imposed to help reduce the possibility of such events recurring.

Publicly listed American companies are now subject to the Sarbanes–Oxley Act (SOX) of 2002 [1] to help ensure the accuracy of the data coming from financial systems. Medical device manufacturers are subject to a raft of regulations such as the United States Quality Systems Regulations 21 CFR part 820 [21] or the European Medical Device Directive [19]. These diverse sources of regulations are increasing as software continues to push boundaries, be misused and get embedded in ways which were not envisioned before.

6.2.4 The G-SC Case Study

To examine these aspects in detail we will look at the transformation of G-SC, a global leader in supply chain business process management. From an IT perspective, the company moved from a collection of around 30 international self-sufficient facilities, to a centralised, shared services IT model which included a globally distributed software development team with members located in the United States, Europe and Asia. This transformation was in fact an evolution of a company which grew organically initially and then through acquisitions and mergers. We will examine it from a process evolution perspective where we will also see how external influences, in particular from regulatory bodies, helped to shape that evolution.

6.3 Process Evolution

The structure of an organisation is crucial to its success. While it is important to design an appropriate structure in the early days of an organisation, it needs to be continuously appraised, *"A company needs to continuously revisit and challenge its answers to the who-what-how questions in order to remain flexible and ready to adjust its strategy…"* [30]. However, there is a level of foresight or design beyond which you cannot go with any certainty. Slack and Lewis [29] tell us *"…many environmental and operational variables are unknown in advance (and in some cases, unknowable)"* and therefore it is imperative to periodically review your situation. Thomas Friedman writes in his book 'The World Is Flat' that *"the best companies stay healthy by getting regular chest x-rays"* [22].

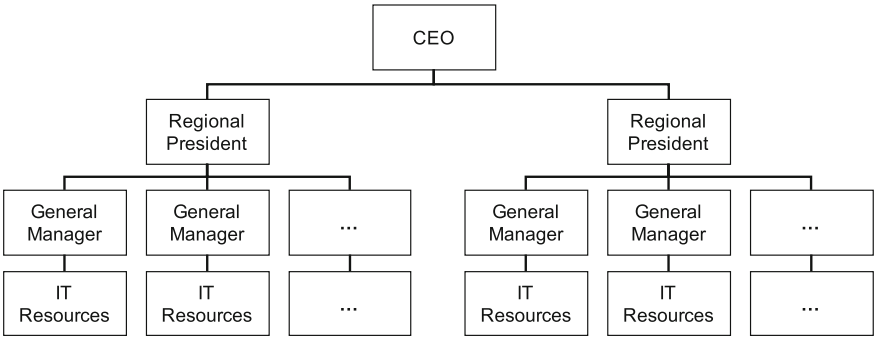

Fig. 6.1 G-SC initial management structure

Hand in hand with structure goes the way in which an organisation operates. Job descriptions are specified, roles are assigned, levels of authorisation are conferred, operating procedures defined and so on. These processes are overlaid on the management structure and are typically closely adhered to. But the world does not stay still, and so successful organisations are the ones most able to adapt to the changing environment in which they find themselves.[1]

Different environments can sustain totally different management structures and it is difficult to know precisely what will and will not work well. But experience shows that change is inevitable. New Chief Executive Officers (CEO) favour different management structures, new Chief Information Officers (CIO) favour certain technologies or development methodologies. Sometimes change is forced through, sometimes the status quo is required before introducing change. For example, when IBM bought Lotus Development in 1995, in the biggest corporate software take over up to that time, the contrast in work styles, dress codes, management hierarchy, etc., was stark, and a large risk to the entire project. IBM was very careful not to march in and take over, but gradually integrated the workforces and technologies over time (resistance is futile).

6.3.1 The G-SC Evolution

G-SC focused on high technology industries and as such was heavily invested in bespoke systems development. Consequently, they had built up a substantial IT workforce which included a large number of software engineering professionals. The initial global management structure is depicted in Fig. 6.1 and shows a typical structure for a disparate collection of self-sufficient solution centres (focus here on IT resources).

[1] I borrow the phrase from a quote about the evolution of species often attributed to Charles Darwin. There is some debate about whether Darwin actually said or wrote these exact words.

The individual General Managers (GM) had a lot of autonomy and full control over all local IT resources which allowed them build up a strong reputation in the local market for excellent customer service and rapid response times. Typically each solution centre had an IT manager with support, business analysis and software development expertise. This provided them with everything a good software engineering function needs.

The software engineering function within G-SC could be classified as residing somewhere between packaged and IS groups. Carmel and Sawyer [8] believe that *"…packaged software firms function in an environment of intense time-to-market pressure relative to IS development efforts"*. However, G-SC (clearly not a Packaged type organisation) was expected to have new business processes operational within the timelines governed by the customer (often packaged software firms) who in turn operated to their own specific market-driven product release schedules or seasonal consumer activities. Thus, having full control of local resources meant they could respond to these demands by reassigning resources, reprioritising projects or a combination of the two.

6.3.2 Sharing the Service

G-SC went through a merger and a subsequent acquisition which brought large-scale changes to the organisation. Along with a change in the business strategy, for example building global business teams to service global customers, the company introduced the role of CIO. The intention was clear, look for ways to gain efficiencies in the IT systems and personnel. This meant some form of integration from both perspectives, not a task to be underestimated [14]. From a globalisation perspective, in order to achieve integration, some level of standardisation is required, but, according to [8] and as we found out, the effort for standardisation of packaged teams pales in comparison to the scale of obstacles that a global IS function has to deal with.

By grouping all IT resources under the CIO, the new organisational structure looked like Fig. 6.2. Beneath the CIO, all IT resources were sub grouped into broad areas with further sub groupings for more specialised functions like software development. The immediate and involuntary effect of this reorganisation is that the new "Global" software development group becomes a bottleneck to the organisation since every solution centre vies for the finite resources (as indicated by the direct lines). But this reorganisation was in response to the new business strategy, to present the company as a unified, focused and globally serving service organisation.

It is worth noting that aligning the business and ICT strategies is a never ending cycle. *"(IT alignment) is complex, multifaceted and never completely achieved. It is about continuing to move in the right direction and being better aligned than competitors. This may not be attainable for many enterprises because enterprise goals change too quickly, but it is nevertheless a worthwhile ambition because there is real concern about the value of IT investment"* [26]. The CIO in this case has responsibility for ensuring that alignment.

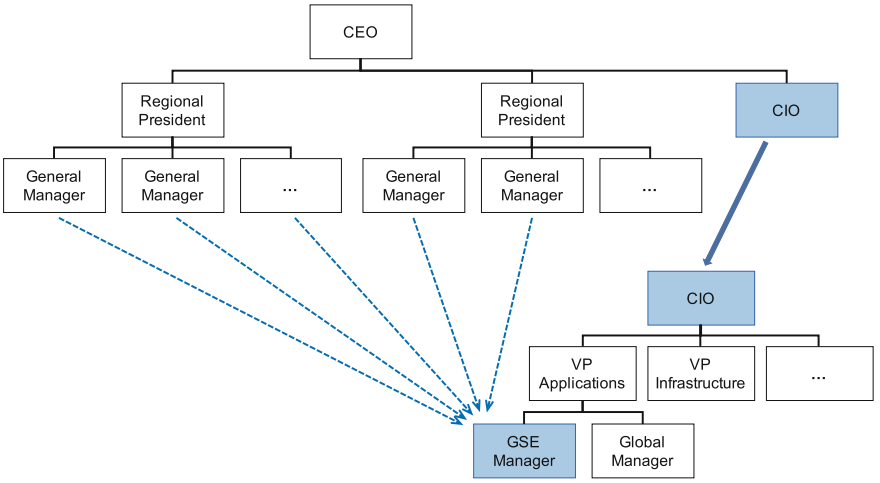

Fig. 6.2 The new G-SC management structure

6.4 Some Growing Pains

Operating as a shared service means that the group gets software project requests from all corners of the organization. In a global context that means that internal customers can be physically located anywhere and also that different business managers and indeed regional presidents are vying with each other for development resources. Many times this led to conflict and internal management escalations in order to secure resources. Sometimes these conflicts required resources to be sourced through expensive external consultants. Resourcing a project by this means has the benefit of a quick solution but also tends to incur a large amount of technical debt (see Chap. 15).

6.4.1 Project Prioritisation

One of the most problematic areas with this new structure was around the scheduling and prioritisation of software projects. With a finite number of GSE resources, and disparate business units needing work done, how do you decide on the order in which projects will be scheduled? Each manager can equally claim that their project is critical. To remediate this, a project review board (PRB) was instigated which consisted of a business representative from each solution centre attending a weekly conference call and helping set the priorities of development projects for the region. Key IT personnel who could advise from a technical perspective also attended these calls (Fig. 6.3). It is important to say that the business representatives had to have the authority to speak on behalf of their centres, which is why it was crucial that the respective General Manager appointed them to the PRB. Overall this worked

Fig. 6.3 The software review board structure

quite well but it still proved difficult to get consensus on prioritisation when multiple centres were under pressure to deliver projects within the same timelines.

Educating the internal business community was therefore also required and was performed by means of global email communications and solution centre visits. It is also worth noting that having management located in Ireland (a "Bridge") did help alleviate some of the temporal distance issues, since normal working days overlapped between Asia and Ireland and also the US and Ireland [31].

6.4.2 Personnel Management

From a software personnel management point of view, a number of issues arose. As the manager of a new GSE function, you inherit a lot of resources spread out across the globe. Personal unfamiliarity, time zone differences, cultural differences, varying levels of expertise all make for a very dynamic environment. Some people are better than others at adjusting to such a new working environment. Many people see this as a threat to their position, status, or very job, and so the management approach requires some significant adjustment.

The primary objective should be to get the group familiar with each other and comfortable working together. A key manifestation to overcome in the team is one of fear [10]. Temporal issues are extremely difficult to eradicate completely, but implementing processes around working arrangements can assist. For example, at times European developers worked the equivalent of US times to keep a project on track. Due to an asymmetry in knowledge and skills it took a long time before a more "follow the sun" approach could be implemented. Issue resolution was on average longer when dealing in the distributed environment but specific escalation paths were introduced in order to expedite special cases.

A source of much frustration for the software developers was getting in touch with someone at a remote site. This was typically to help with things like clarifying user requirements, user-testing functionality or carrying out a local installation. The PRB process, however, ensured that each site had a representative who could help get such situations resolved swiftly.

Building a team ethos through regular communication, sharing knowledge, cross-site projects, site visits and perseverance are some of the tools in the manager's toolbox for making this work. Building relationships with remote groups and aligning with other functions to deliver a coherent service are also things that need to be undertaken. For example, within a distributed organisation you will typically find local technical support personnel who act as the first line of support for local system issues. These people need to be considered part of the GSE process and should be an integral part of any system deployment, with knowledge transfer sessions and supporting documentation made available. Coordinating such handover processes needs careful consideration [10].

6.4.3 Seeing the Wood Despite the Trees

Following any organisational transformation it is easy to point out the hardships endured and the failures which occurred. But being part of a multi-national, multi-cultural, cross-functional team of skilled professionals is a fascinating, educational and often exhilarating experience. Fundamentally people are the same. They have similar concerns, ambitions, dependencies, and fears. But once things settle down and they start seeing global projects, of which they played a part, being deployed successfully, it generates a sense of satisfaction and belonging to something bigger, that they otherwise would not have been exposed to.

In this sense, whole teams of people evolve in their roles as the organisation evolves. People start to think outside their own box. They start to ask questions about the possible usefulness of a particular local process or project to the wider organisation. They see opportunities to share what they do but also the potential of using what other people do. For example, in G-SC a production manager may come up with an idea on how to better organise a certain manufacturing process but which requires a specific piece of functionality to be developed. The global software engineer/business analyst who examines this request starts to see how this might also work in other manufacturing sites around the world. By a slight extension/modification to the original requirements a globally reusable component is developed and made available to anyone who wants it.

It is imperative to recognise these successes [28]. People like to see that what they are doing is contributing to the success of the larger team. It helps to build or simply maintain employee motivation. It helps build confidence in the new process.

6.4.4 Regulating the Software Process

Software regulations typically expect an organisation to have a published software development process which clearly shows how the concerns of the regulator are addressed. In addition, the organisation will have to prove that they are following this

approved process through some form of record keeping and usually an external audit. Interestingly there is much debate about the suitability of the more modern software development methodologies such as agile or lean in such regulated domains [13]. For more information on agile and lean approaches to Software Engineering, see Chap. 2 and Chap. 3.

Due to differing concerns between domains, different software development regulations have been created and consequently affect the development processes differently. The SOX regulations, for example, were designed to ensure accurate financial reporting. Very different concerns, however, are at the heart of regulations pertaining to safety-critical domains such as Aviation, Nuclear, or Medical Devices. In such domains the obvious concern is for human safety, and the regulations are designed to minimize to an acceptable level any related risk. To be SOX compliant you are expected to demonstrate that the software has been adequately tested. However, safety-critical regulations put a much heavier emphasis on this and expect thorough verification and validation of the software [20]. This level of detail, as required by the international standards such as IEC 62304 [25] for Medical Devices software life cycle processes, and ISO 13485 [24] for Medical Devices quality management systems, is far more onerous, and compliance must be demonstrated right throughout the entire software development lifecycle.

6.4.4.1 Regulatory Effects on the GSE Function

The SDLC within a regulated area is reflective of a number of key influences. My prior research [11] categorises these influences into 4 groups (Fig. 6.4). Within the model, the arrows emanating from the Regulations box are shown leading to the other three categories, indicating that compliance with the regulations must be addressed within multiple levels and contexts.

The organisation component influences the software development process by defining development tasks and delegating roles (such as developers and testers), responsibilities (such as project management and software validation), and authority (such as approvals). The regulatory context will influence the organisation's ethos

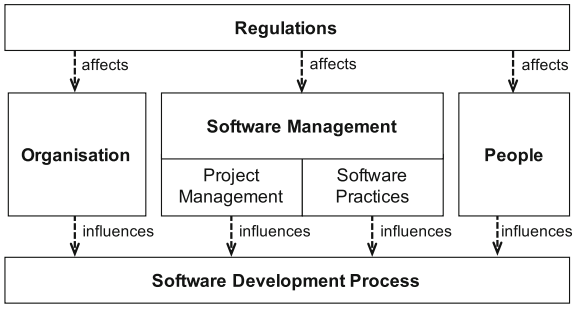

Fig. 6.4 Categories of influences on the software development lifecycle within a regulated context

in terms of ensuring the quality of the software, the workforce's attitude to risk management, and their sense of responsibility.

The software management component (of tasks, resources and schedules) naturally influences the SDLC since it will be within these competencies, capabilities and situational contexts that the SDLC will need to be framed. For example, the technical nature of the product will automatically dictate the type of skills required and the type of development environment needed. The availability or lack or availability of these resources will shape the resulting SDLC [27]. Different people and organisations approach project management of software development differently. A company which sees the software as being strategically important may, therefore, also be more supportive of pursuing software process improvement initiatives.

The effect of regulatory compliance is very notable at a personnel level as it is precisely the human activities that are being governed. When moving from an unregulated into a regulated environment, unless the work processes are already fulfilling the regulations (experience suggests that this is unlikely), there is a need for peoples' daily activities to change. For example, both SOX and the Medical Devices regulations look for some level of independence in certain key areas. SOX looks for segregation of duties when it comes to code deployment or even access to a production system, while Medical Devices regulations expect independence between developers and validation engineers. The typical approaches to activities such as communication and knowledge transfer, where important ad hoc conversations go undocumented, or an approval is given verbally, are no longer acceptable. When people are accustomed to operating in an environment where issues can be fixed "on the spot," these tighter controls can be very frustrating for both the technical employee as well as for the person awaiting resolution.

6.4.4.2 G-SC under SOX Regulations

In 2002, the SOX regulations were passed into law in the United States. A critical element of those regulations refers to the ITGCs (Information Technology General Controls) which are intended to ensure: that financial data is stored securely, that only the relevant people have access to certain systems and functionality, and also that any software/modifications developed which could affect the financial data are developed within a robust and documented software development process.

It was therefore imperative that, within the G-SC environment, management were confident that each developer, regardless of location, adhered to the internal processes which were aligned with the expectations of SOX. A single set of "Global" processes/controls was rolled out to all developers and support members and proved instrumental in moving toward a cross-regional function. However, it added an extra overhead to management, who then spent a substantial amount of time ensuring that all team members, especially external contractors, were following the processes and maintaining the necessary documentation.

In addition, the regulatory requirement for segregation of duties is worth noting. For example, a software developer should not have full access to a live production

system. Prior to being regulated, the G-SC developers were given such access by default. This was in order for them to perform technical tasks such as code deployment and resolving production issues. To mitigate the regulatory concerns, the database administrators were trained up in how to perform the deployments and where necessary supervised by the software expert. For trouble shooting production issues, the software engineers were given temporary administrator access, a record kept of when and why it was granted, and access revoked upon issue resolution. These records would subsequently be reviewed during the annual SOX audit.

6.5 Conclusion

As a company changes over its lifetime, for whatever reason, it must modify its business processes accordingly. The software engineering function which supports these processes must therefore be amenable to change also. The larger the company, the more disruptive the change but perhaps the more necessary it is.

Contemporary businesses are under huge pressure from competitors. In particular smaller companies must fight hard to win business from larger companies. Large companies are often looking for ways to reduce their costs in order to remain competitive. From a software engineering perspective, both small and large companies are looking at having some of their software developed at remote locations. Small companies are looking to outsource to low cost economies, while large companies are looking to "virtualise" their distributed software engineers into a globally shared service. In both cases, this new GSE function cannot work the same way as before. The process must change to support the business.

In addition, regulatory controls which are becoming more pervasive within the software engineering function, introduce additional complexities to the smooth delivery of the GSE service. The effects are felt right throughout the SDLC, and are magnified when the context is a global organisation.

With change comes challenge and opportunity. The challenges are not insurmountable and the opportunities reveal themselves as you evolve. We need to overcome the first to exploit the second.

6.6 Further Reading

The GSE topic has undergone quite a lot of academic research in the last number of years. The seminal work by Erran Carmel "Global software teams: collaborating across borders and time zones" [7], is a must read for those wishing to delve in to the subject. A research team based in the University of Limerick, Ireland have spent the last 10 years researching the theme, and have developed a global teaming best practice model. Some useful papers include: [9, 17, 32]. Readers are also encouraged to peruse their website at http://www.lero.ie/publications.

As introduced in this chapter, there has been a large growth in the number of industries which are becoming subject to software regulation, for security, criticality, safety, or quality reasons. This is still an area which requires further examination. The question remains as to how the current GSE research is relevant in these regulated contexts. Some introductory readings include: [6, 15].

References

1. 107th Congress: Sarbanes-Oxley Act of 2002. Technical report. Enrolled Bill: H.R. 3763, Congress of the United State of America (2002)
2. Ågerfalk, P.J., Fitzgerald, B., Holmström, H., Lings, B., Lundell, B., Conchúir, E.Ó.: A framework for considering opportunities and threats in distributed software development. In: Proceedings of the International Workshop on Distributed Software Development, pp. 47–61. Austrian Computer Society (2005)
3. Bank for international settlements: Basel II: international convergence of capital measurement and capital standards: a revised framework. http://www.bis.org/publ/bcbs107.htm (2004)
4. BBC: enron scandal at a glance. http://news.bbc.co.uk/2/hi/business/1780075.stm (2002)
5. Borrás, C.: Overexposure of radiation therapy patients in panama: problem recognition and follow-up measures. Pan Am. J. Public Health **20**(2–3), 173–187 (2006)
6. Campbell, M.: Regulations. IEEE Potentials **23**(2), 14–15 (2004)
7. Carmel, E.: Global Software Teams: Collaborating Across Borders and Time Zones. Prentice Hall, Upper Saddle River (1999)
8. Carmel, E., Sawyer, S.: Packaged software development teams: what makes them different? Inf. Technol. People **11**(1), 7–19 (1998)
9. Casey, V., Richardson, I.: Practical experience of virtual team software development. In: Proceedings of the EuroSPI 2004 Industrial Proceedings. Trondheim (2004). http://ulir.ul.ie/handle/10344/2149
10. Casey, V., Richardson, I.: Virtual teams: understanding the impact of fear. Softw. Process.: Improv. Pr. **13**(6), 511–526 (2008)
11. Cawley, O.: The application of a lean software development methodology within the regulated domain of medical device software. Ph.D. thesis, University of Limerick (Computer Science and Information Systems) (2013)
12. Cawley, O., Richardson, I.: Lessons in global software development – local to global transition within a regulated environment. In: European Systems and Software Process Improvement and Innovation (2010)
13. Cawley, O., Wang, X., Richardson, I.: Lean/agile software development methodologies in regulated environments - state of the art. In: Abrahamsson, P., Oza, N. (eds.) Lean Enterprise Software and Systems. Lecture Notes in Business Information Processing, vol. 65, pp. 31–36. Springer, Heidelberg (2010)
14. Cawley, O., Wang, X., Richardson, I.: Regulated software development - an onerous transformation. In: Weber, J., Perseil, I. (eds.) Foundations of Health Information Engineering and Systems. Lecture Notes in Computer Science, vol. 7789, pp. 72–86. Springer, Heidelberg (2013)
15. Cawley, O., Richardson, I., Wang, X., Kuhrmann, M.: A conceptual framework for lean regulated software development. In: Proceedings of the 2015 International Conference on Software and System Process, pp. 167–168. ACM, New York, USA (2015)
16. DeLone, W., Espinosa, J., Lee, G., Carmel, E.: Bridging global boundaries for is project success. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, p. 48 ff. IEEE Computer Society, Washington, DC (2005)

17. Deshpande, S., Beecham, S., Richardson, I.: Global software development coordination strategies - a vendor perspective. In: Kotlarsky, J., Willcocks, L., Oshri, I. (eds.) New Studies in Global IT and Business Service Outsourcing. Lecture Notes in Business Information Processing, vol. 91, pp. 153–174. Springer, Heidelberg (2011)
18. Duranton, M., Black-Schaffer, D., De Bosschere, K., Maebe, J.: The hipeac vision for advanced computing in horizon 2020 (2013)
19. European Union: Medical Device Directive 2007/47/EC of the European Parliament and of the council. Official Journal of the European Union (2007)
20. FDA: General Principles of Software Validation; Final Guidance for Industry and FDA Staff. FDA Standard, U.S. Food and Drug Administration – Center for Devices and Radiological Health (2002)
21. FDA: Code of Federal Regulations 21 CFR Part 820 - Quality System Regulation. FDA Standard Part 820, U.S. Food and Drug Administration (2015)
22. Friedman, T.L.: The World Is Flat: A Brief History of the Twenty-First Century. Holtzbrinck Publishers (2005)
23. IEEE Reliability Society: Annual technical report 2008. Transactions on Reliability **57**(3), 398–425 (2008)
24. ISO: Medical devices – quality management systems – requirements for regulatory purposes. International Standard ISO 13485:2003, International Organisation for Standardisation (2003)
25. ISO/TC 210: Medical device software – software lifecycle processes. International Standard IEC 62304:2006, International Standards Organization (2006)
26. IT Governance Institute: Board briefing on it governance. Available from http://www.isaca.org/Knowledge-Center/Research/ResearchDeliverables/Pages/Board-Briefing-on-IT-Governance-2nd-Edition.aspx (2003)
27. Kettunen, P., Laanti, M.: How to steer an embedded software project: Tactics for selecting the software process model. Information and Software Technology **47**(9), 587–608 (2005)
28. Kotter, J.P.: Leading change: why transformation efforts fail. Harvard Business Review **73** (1995)
29. Lewis, M., Slack, N.: Operations Strategy. Prentice Hall, Upper Saddle River (2002)
30. Markides, C.C.: A dynamic view of strategy. Sloan Manag. Rev. **40**(3), 55–63 (1999)
31. Richardson, I., Avram, G., Deshpande, S., Casey, V.: Having a foot on each shore - bridging global software development in the case of smes. In: Proceedings of the International Conference on Global Software Engineering, pp. 13–22. IEEE, Washington, DC (2008)
32. Richardson, I., Casey, V., Mccaffery, F., Burton, J., Beecham, S.: A process framework for global software engineering teams. Inf. Softw. Technol. **54**(11), 1175–1191 (2012)
33. The Free Dictionary: regulations. http://www.thefreedictionary.com/regulate (2015)