Caught disaster: using the layered systems model as a diagnostic tool for wayward software projects

Phil Robinson and
Martin Ringer

The authors

Phil Robinson has worked with information technology, in a variety of roles since 1975. He is an experienced workshop facilitator and has presented numerous training courses for organisations in Australia and SE Asia.
Martin Ringer has been an independent management consultant in Perth, Australia, for eight years. He has extensive international experience in enhancing team effectiveness, working with severely conflicted teams and in training consultants and group workers.

Abstract

This paper introduces The Layered Systems Model and describes how it can be applied to assist project managers to be effective at all stages of project design and delivery. Conventional rational problem solving techniques for difficult projects are supplemented with imaginative and relationship-focused approaches. The Layered Systems Model is a management tool for focusing on different levels of functioning in projects, organisations, and teams. The model acts like a series of "filters to perception" to assist managers to assess how well an organisation is functioning at each level. It can be applied to large organisations, departments, and small work groups or in the case of this paper, software project teams. The paper shows how the Layered Systems Model can happily coexist with the Project Management Institute's eight project knowledge areas to form a powerful diagnostic tool for software projects which are not quite "out of control" but are heading in that direction.

Electronic access

The current issue and full text archive of this journal is available at
http://www.emerald-library.com

Introduction

We invite you to join us on a journey into the surprising and colourful territory of the imagination, of imagery symbolism. This exploration has a purpose, and will end in familiar places. So for now, suspend your judgement, and begin

If the last software project you were involved with had the character of an animal, what sort of animal would it be? Before you become fully immersed in this paper we invite you to pause for a minute and reflect.

Would it be a tiger, an elephant, a jellyfish, a crocodile, a pig . . .? What is the nature of this animal? Is it friendly, aggressive, sub-missive or something else? What does it look like, smell like and sound like? What do you tell yourself about it? Depending on how your last project went and how you felt about it, a particular animal probably sprang quickly into your mind. Associating a software project with an animal provides a powerful and at times amusing symbol for the project but how do we choose the animal that feels right? What mechanism are we using when we take a team of people, all focused on creating a system from complex technology and associ-ate their endeavours with a spotty dog? Is there a way that we can harness this uncon-scious process to help us better manage our software projects? We think there is and we explore this possibility under the banner of the term (derived from Jungian ideas) of the "Archetypal role."

Archetypal roles

Consider what comes to mind when you think about how you should handle your project animal? What does this tell you in regard to the way you feel about the project? Do you approach the project, chair in one hand and whip in the other as if taming a tiger or do you simply stand in your gum (Wellington) boots and empty a bucket of swill into the project? Does your animal need special care and attention? Will it bite you if you let your attention lapse? Where does your animal live? In the jungle, a zoo or on a farm? What does it eat? Is it a predator, preying on other unsuspecting animals or a herbivore quietly chewing the cud?

Letting your imagination run wild with these associations may help you to understand

the way you really feel about the project. Don't limit yourself to animals as symbols. Try expanding the possibilities to pop songs, operas, paintings, books, movies, sitcoms … the range of possible symbols is huge. Playing with this idea of a symbolic representation of the project, you could ask other project team members for their symbols. That may help both them and you to understand their real feelings about the project. It might help the whole project team develop a shared sense of the "project in the mind" (Morgan, 1997; Shafer, 1999). Both commonalties and differences between team members can be surfaced and talked over. Bear in mind, though, the conclusions that you draw may or may not be aligned with the official project propaganda.

We like to describe the association of projects with animals and other symbols such as songs, films, places or cars, as the project's "archetypal role". Another way of viewing these associations is that they are "mental models" (Senge, 1992), or aspects of "the project in the mind". The mind is a place where anything is possible but where "the possible" is heavily coloured by our values, experiences, knowledge and beliefs.

The inner self

What we call the "inner self" is a place in which we are normally careful about what we present to the outside world. While it is quite acceptable to playfully ask a colleague what animal he or she associates with a project, it is not always acceptable to inquire about a colleague's deeply held attitudes, beliefs, values – even if you are focusing primarily on their personal response to being involved in a particular project. However, these values, impressions and feelings can have a profound effect on the success or failure of a project. If an individual's values do not fit with the project they will have to work hard to "appear" to fit. If the project is a tiger staffed by a tiger team then a well-meaning goldfish will not fit in. Conversely a project staffed by wise old owls may not welcome an inquisitive monkey into their team.

It is well known that an individual's skills, personality, emotional health and fit to the project will have a major influence on a project. McConnell (1996) nominates "undetermined motivation" and "uncontrolled problem employees" as two of the classic mistakes of the software industry. In his book *Software Runaways*, Glass (1998) identifies "the lack of appropriate people to do the job" as a factor in runaway projects.

However, many of these issues are intensely personal. While it is acceptable to objectively assess a team member's experience and skills, it is not acceptable to deeply probe their inner selves in an uninvited fashion (Ringer and Gillis, 1995).

Frequently, the first indication that something is amiss will be picked up from their performance in relation to the project. When questioned about a change in work performance, a team member may directly offer a personal problem as the reason, or they may provide what seems like a vague, unsatisfactory explanation. Aggressive probing of these vague answers may produce extreme discomfort in the team member. This may or may not be visible to or noticed by their interrogator.

Unless our colleagues volunteer information about their inner selves, exactly what is going on inside their minds will remain, in most cases, behind a locked door. Unless we have access to skilled therapeutic or organizational consultancy we must resort to observing the way they present themselves to the outside world. We call this the "interacting self".

Interacting self

Focusing on the interaction between members of a project team enables you to identify their ability to carry out the technical work, the work of managing relationships inside the team and the work of relating to the clients for whom the project is being conducted.

Effective project managers will widen their scope of their interest beyond the internal workings of the project team to include the clients for their software. It could be enlightening to identify the kind of animal that a particular client reminds you of. From here you might muse about the intensely personal issues that they face, and to use this information to help you interact with them in a more constructive way.

Interaction is also about communication, and as any communications engineer worth their salt will tell you, communication needs a medium to carry it. The relationship between

people is the medium that carries their messages when they are communicating. When the relationship is healthy, clear communication can take place. When there are problems, the relationship will distort and corrupt the messages. So important is this fact, that we would go as far to say that quality, co-operative work can only occur when the relationships between people are good. In the absence of good relationships, people withdraw into their inner worlds and "do their own thing". Software projects provide more than ample opportunities for people to withdraw. The nature of the work means that people who have difficulty relating with others can exist for long periods of time in solitary relationships with their workstations. This can produce outstanding results in small projects but as soon as co-operation, and its close relative communication, are required things will start to fall apart.

The interacting self also focuses on how individuals interact with their work environment and the software tools, facilities and workstations that they use. In a classic study, DeMarco and Lister (1987) found a strong correlation between office space per employee and productivity. In fact it was the strongest correlation amongst all the factors which they studied. Many project managers will already feel on safe ground ensuring that team members have desks, phones, software tools and work stations so we only mention these interactions in passing.

Returning to relationships as the carriers of communication, it is obvious to most people that there are two types of relationship encountered in project teams. Formal, documented relationships are established between the various project participants as part of the project planning process. The roles and reporting structure of the programmers, analysts, designers and managers involved in the project are usually well defined. Alongside these formal relationships, informal relationships with friends, enemies, mentors and rivals spontaneously come into being.

Informal structure

People bring existing informal relationships with them to a new project and they quickly form socially based informal relationships with others when they join a project team. These informal relationships may or may not support the formal structure and the ability of the team to carry out its task. Furthermore, people not directly involved in the project may be informally conscripted to help via existing relationships, networks and contacts. These invisible others may be an essential element of a successful team but may not be visible to any but the insiders (Egan, 1994).

Informal power blocks, "gangs" and "self-help groups" inevitably form in a project of a reasonable size and duration. In longer projects they can become institutions that develop their own stories, myths and culture. In particular, subgroups form around team members' formal qualifications and their preferences for specific brands of software and hardware. These informal groups may hinder or assist the achievement of the project's objectives because they work at a clandestine level to keep their own informal group in the pole position of influence (Standing, 1998).

We call the network of informal relationships associated with a project the "informal structure". The informal structure is not restricted to just relationships between people. It also includes any part of the project infrastructure that is put in place informally without conscious choice or effort on the part of the project management. Homespun and *ad hoc* solutions will be sucked into the vacuum created by projects that don't have formal quality plans, change control mechanisms, budgets or proper schedules.

Typical software projects appear to utilise a lot of informal structure when compared to other disciplines. In the worst cases, the actual project scope itself is specified informally, although research shows that such a project is highly likely to fail (Glass, 1998). In some cases, the project managers themselves are responsible for creating informal structures. Jones (1991) estimates that in the USA 50 per cent of software projects have "irrational schedules". He defines an irrational schedule as one where a "pre-determined end date is selected and it is forced on the project by arbitrary decree". Jones also estimates that anything up to 30 per cent of software development in the USA and 50 per cent in Japan is informally funded by unpaid overtime. This "informal effort" is usually not recorded with the result that producing meaningful comparisons or measures of productivity is almost impossible.

DeMarco (1982) suggests that informal time recording is a widespread practice, "time is not charged against a task until everyone is confident that charging the time will have no adverse political effects". So where does the time get charged? DeMarco offers an amusing scenario. "The worker looks for someplace else to hide the time: 'Why, here's a charming possibility' Task 45-6792.4, INVESTIGATE USER SECONDARY PREFERENCES, a task with 260 remaining hours".

Programmers love to create informal, homegrown software tools. In one "runaway" project described by Glass (1998), the programmers created a software tool that allowed them to "bypass" the formal configuration management system and check their code in and out of the repository at will. Their intentions were not malicious, they were simply responding to the unfair manner in which their performance was being measured. Needless to say, there were disastrous consequences arising from this informal shortcut.

Informal structures will always exist in any software project. In many cases, they can be powerful, positive forces that can assist the project to meet its goals. Project managers tamper with healthy informal structures at their peril! In other cases, informal structures exist because the necessary project infrastructure has not been put in place by the project management. When a project's infrastructure is consciously planned, and implemented we call it "formal structure".

Formal structure

The formal structure of a project includes artefacts such as a statement of the project's scope, estimates of effort, work assignments, schedules, budgets, supplier contracts, quality plans, and the project reporting structure. These and other aspects of the formal structure are identified and described in the Project Management Institute's (1996) *A Guide to the Project Management Body of Knowledge.* The Project Management Body of Knowledge (PMBOK) identifies nine discrete areas worthy of a project manager's attention:

(1) *Project integration management* describes the processes required to ensure that the various elements of the project are properly co-ordinated. It consists of project plan development, project plan execution, and overall change control;

(2) *Project scope management* describes the processes required to ensure that the project includes all the work required, and only the work required, to complete the project successfully. It consists of initiation, scope planning, scope definition, scope verification, and scope change control;

(3) *Project time management* describes the processes required to ensure timely completion of the project. It consists of activity definition, activity sequencing, activity duration estimating, schedule development, and schedule control;

(4) *Project cost management* describes the processes required to ensure that the project is completed within the approved budget. It consists of resource planning, cost estimating, cost budgeting, and cost control;

(5) *Project quality management* describes the processes required to ensure that the project will satisfy the needs for which it was undertaken. It consists of quality planning, quality assurance, and quality control;

(6) *Project human resource management* describes the processes required to make the most effective use of the people involved with the project. It consists of organisational planning, staff acquisition, and team development;

(7) *Project communications management* describes the processes required to ensure timely and appropriate generation, collection, dissemination, storage, and ultimate disposition of project information. It consists of communications planning, information distribution, performance reporting, and administrative closure;

(8) *Project risk management* describes the processes concerned with identifying, analysing, and responding to project risk. It consists of risk identification, risk quantification, risk response development, and risk response control; and

(9) *Project procurement management* describes the processes required to acquire goods and services from outside the performing organisation. It consists of procurement planning, solicitation planning, solicitation, source selection, contract administration, and contract close-out.

Most project managers would agree that it is not necessary to create a such a comprehensive formal structure for each and every project. However, by the same token, project managers should be aware that if they fail to define a formal structure for one of these areas and it turns out to be an important issue during the life of the project, an informal structure will spring up to meet the need. That is, informal (and hence potentially invisible) structures will spontaneously emerge in project teams if the formal structure is inadequate in any key areas.

So far in this paper we have seldom referred to the actual nature of the work conducted by project staff. That is the activities that are usually seen as the "real work" of the project. The formal structure (above) provides a means of allocating these activities to people and of providing appropriate resources, authority and information to team members to enable them to carry out the activities involved in delivering the project. However, no project can be effective unless the activities carried out in its delivery are appropriate to the client's needs or request. Focus on the "process" level enables the necessary activities to be identified.

Process

Process is inextricably linked to project scope. The overarching statement that encompasses all activities is what we call the project "primary task" (Miller, 1993) which outlines the purpose of the project and the means by which it will be implemented. Every activity carried out in the implementation of this primary task should directly lead to the production of the final product, or assist with providing the informal and formal structures to support the project. The PMBOK names these two different types of process as:
(1) Product-oriented processes; and
(2) Project management processes.

In the case of software projects, the product-oriented processes are defined by the software development methodology that is utilised throughout the project. A word of caution here; in many software projects there is no formal, documented process for producing the product. In these cases, the vacuum created will be filled by an *ad hoc*

methodology that forms a part of the informal structure.

The PMBOK (Project Management Institute, 1996) further provides a breakdown of the project management processes into the following groups:
- initiating processes;
- planning processes;
- executing processes;
- controlling processes; and
- closing processes.

In our experience, the key success factor in the process level is that of successfully defining with the client the primary task. That is, the client-consultant system is in agreement about the purpose of the project and the fundamental means by which it will be carried out.

The layered systems model

This paper has taken us on a journey that has covered a lot of ground. We have visited the mystical and symbolic, the emotive and personal, and then the social and interpersonal, finally arriving at the supremely logical and procedural. On the way we have moved from a focus that was primarily "right-brain" to one dominated by "left-brain" functioning. Thank you for staying with us. In recognition of your endurance, we offer a souvenir map of the territory we have covered. We call it the Layered Systems Model (Figure 1). The model is more than an ornament; it is a useful tool that you may wish to apply in your day-

Figure 1 The layered systems model
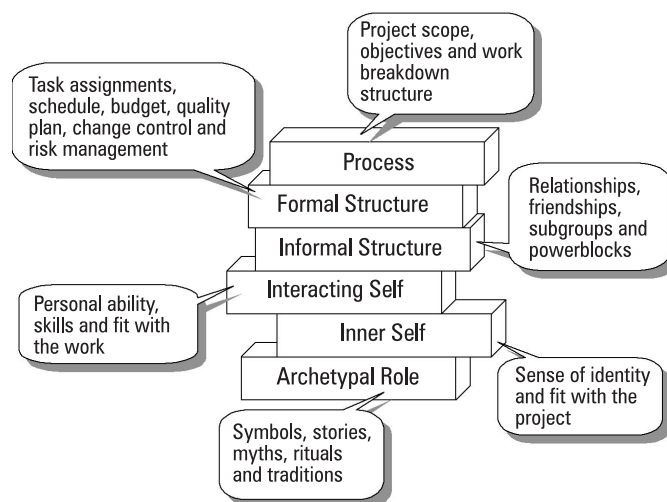
to-day work to enhance the effectiveness of projects with which you have involvement.

The Layered Systems Model identifies six layers of concern in a software project. It can be used as a set of "mental filters" to bring into focus different aspects of a project one at a time. Selecting the correct filter, should reveal the problems and issues at each of the individual layers. These can be explored and analysed. Armed with their analysis, project stakeholders are in a much better position to take informed action to correct the situations they encounter as opposed to relying on their intuition or "gut feel" to "fly by the seat of their pants".

Applying the model to projects

Here, we offer some practical suggestions about how to apply the Layered Systems Model to the management of software projects. The first step is to "diagnose" or to identify where your energy should be focused. The diagnosis provides pointers for action. In its simplest form this diagnosis can consist of six questions (Ringer and Robinson, 1996). Each question corresponds to one of the layers.

The six questions are:
(1) Is the scope of the project identified, articulated and agreed?
(2) Are the structures in place to allocate the key tasks, resources and responsibilities?
(3) Are the networks of relationships in the project assisting the project to achieve its objectives?
(4) Is each team member competent to carry out their assigned project tasks including the task of managing relationships between themselves and others?
(5) Is there adequate fit between each team member and the demands of the project?
(6) Is the project driven by or supported by a widely shared and deeply held sense of inspiration and identity and are all of the key players linked with this inspiration at a personal level?

The questions above are only lead-in questions. Answers to these six questions will provide pointers to which level of the project culture requires your most urgent attention. Each of the six questions would be followed by subsidiary questions that expand on the

theme that was opened by your first question at each level.

In practice it is unlikely that you would ask the questions in a tidy sequence as shown above. We suggest that you use the list of six questions above as a reminder of the themes that need to be explored. No project manager will honestly be able to provide unreserved, positive responses to all of the questions, but serious problems in any level indicate a need for action. Further ideas can be derived from the original paper on the Layered Systems Model (Ringer and Robinson, 1996).

Before rushing in to fix the problem, do a thorough check to ensure that there is not a hidden problem at one level that is appearing as a symptom in a different level. For example, an absence of clarity about the project scope may lead to inadequate formal structures, but the dysfunction may be most apparent in the informal layer, with team members fighting over who should do which task. If you attempt to "solve" the problem by working at the level of informal structures – say by running team building exercises – you may achieve temporary relief but the problem will quickly reappear elsewhere.

As a general principle, a change at any level should be checked to identify the impact of that change on the project at other levels. This applies regardless of whether you or someone else originated the change. A further guide to project managers is that the levels at which emotion, story and symbols are most important tend to be the ones that are least adequately managed in today's environment. Pay particular attention to these levels and if you feel out of your depth then ask for help from your peers, others in your organisation, external consultants or other professionals.

Disaster caught or not?

Some software projects fail. These failures cost money, and in the knowledge of the two authors alone, a total of over $15 million has been wasted by four different abortive software projects in Western Australia during the past five years. The Layered Systems Model does not guarantee a panacea, but careful focus on the process level and the level of formal structure can relatively easily preclude avoidable disasters. A comprehensive study of "runaway" software project in the UK undertaken by KPMG (Glass, 1998) revealed

that the top six causes for project failures were:

(1) Project objectives not fully specified.
(2) Bad planning and estimating.
(3) Technology new to the organisation.
(4) Inadequate/no project management methodology.
(5) Insufficient senior staff on the team.
(6) Poor performance by suppliers of hardware/software.

All of these causes could be related to the process and formal structure layers of the Layered Systems Model. So, careful focus on the areas described in the PMBOK (Project Management Institute, 1996) can provide essential preventative risk management for software projects.

However, the studies quoted above have focused on unsuccessful projects, presumably because successful projects are less newsworthy or sensational. In our application of the Layered Systems Model in organisations over the past four years we have noticed that significant improvements can be made to organisational culture through a persistent focus on the "lower" four levels – provided that the levels of process and formal structure are already adequately managed. In other words, projects that are already on the road to success through use of rational project management tools such as the PMBOK (Project Management Institute, 1996) can be turned into outstanding projects with systematic focus on non-rational aspects that is facilitated by use of the Layered Systems Model.

If your project still needs the necessary components of the process and formal structure layers in place, the PMBOK tools are invaluable. Failure to utilise these tools may at best result in your project failing to deliver its objectives, and at worst it will become a "runaway". However, if you have done the necessary groundwork, then the "lower" four levels of Layered Systems Model provide a powerful tool to keep the project on track and tackle the myriad of issues that will inevitably arise during the life of the project.

If nothing else, we invite you to start your next project management meeting by getting everyone to name the animal or another form of symbol that they associate with the project. You may be surprised by how much everyone learns about the project and their own attitude towards it.

References

DeMarco, T. (1982), *Controlling Software Projects: Management, Measurement and Estimation*, Yourdon Press, Englewood Cliffs, NJ.

DeMarco, T. and Lister, T. (1987), *Peopleware*, Dorset House, New York, NY.

Egan, G. (1994), *Working the Shadow Side: A Guide to Positive Behind-the-Scenes Management*, Jossey-Bass, San Francisco, CA.

Glass, R.L. (1998), *Software Runaways: Lessons Learnt form Massive Software Project Failures*, Prentice Hall, Englewood Cliffs, NJ.

Jones, C. (1991), *Applied Software Measurement: Assuring Productivity and Quality*, McGraw-Hill, New York, NY.

McConnell, S. (1996), "Best practices: classic mistakes", *IEEE Software*, Vol. 13 No. 5, (http://www.con strux.com/bp05.htm)

Miller, E. (1993), *From Dependency to Autonomy: Studies in Organization and Change*, Free Association Books, London.

Morgan, G. (1997), *Images of Organization*, 2nd ed., Sage, London.

(The) project Management Institute (1996), *A Guide to the Project Management Body of Knowledge*, (http://www.pmi.org/publictn/pmboktoc.htm)

Ringer, T.M. and Gillis, H.L. (1995), "Managing psychological depth in adventure/challenge groups", *Journal of Experiential Education*, Vol. 18 No. 1, pp. 41-51.

Ringer, T.M. and Robinson, P. (1996), "Focus and strategic action in management: using a systemic model of organisational culture to inform managerial actions", *Work Study*, Vol. 45 No. 6, pp. 5-16 (http://www.mcb.co.uk/services/articles/literati/pdfs/ws456.pdf)

Senge, P.M. (1992), *The Fifth Discipline: The Art and Practice of the Learning Organisation*, Random House, Sydney.

Shafer, A.T. (1999), "Colonial domination and the struggle for identity: a socio-analytic perspective", *Socio-Analysis*, Vol. 1 No. 1, pp. 34-47.

Standing, C. (1998), "Myths and the art of deception in information systems", *Work Study*, Vol. 47 No. 1, pp. 5-13.