Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2000 Proceedings

Americas Conference on Information Systems (AMCIS)

2000

Living Information Systems and Change

Ray J. Pul
Brunel University, ray.paul@brunel.ac.uk

Jasna Kuljis
Brunel University, jasna.kuljis@brunel.ac.uk

Living Information Systems and Change

Ray J. Paul & Jasna Kuljis
Department of Information Systems and Computing
Centre for Living Information Systems Thinking
Brunel University, Uxbridge, Middlesex, UB8 3PH, U.K.
Ray.Paul@brunel.ac.uk Jasna.Kuljis@brunel.ac.uk

Abstract

The Centre for Living Information Systems Thinking (LIST) is looking at IS development in organisations undergoing change. After discussing the problems that change introduces into IS development, the paper explains the basic philosophy of the centre, and some of the areas of research that might make a contribution to the basic questions that have been derived. Some views on the subject of change in organisations will be outlined, and the change paradox will be introduced.

Problems with Information Systems for Business

The first few sections of this paper are a condensation of Paul (1993). There are many forms of information systems (I.S.) in practice. This paper restricts itself to those I.S. that are used by human beings to assist them in their endeavors, the type of I.S. that are exemplified by business I.S. The most generous observation that one can make about current I.S. development is that most, if not all, systems disappoint. This disappointment is mostly with the customer, be that the user or the owner, but also with the analysts and developers. A good example, which demonstrates this point, is the sad case of the London Ambulance Service computer system that spectacularly failed and had to be disconnected on its first days of operation (Watts, 1992). On Monday and Tuesday 26 and 27 October 1992 between 10 and 20 people are alleged to have lost their lives because the emergency service failed them. Whilst such claims are almost impossible to verify, it is clear in this case that the ambulance service customers were 'disappointed'. But what of other interest groups. The man in charge of the system is alleged to have said:

> *"The computer did not fail on that Monday - it was working exactly as it was designed to."*

The report goes on to observe that the people using the computer system - in the control room and in the ambulances - are "doing things wrong". That is, they are not doing what the computer has been designed to expect.

There are many reported I.S. problem developments. The question that one might ask is "are there any complete success stories?" This sometimes gets turned into the opposite question "why don't systems do what people want?" But perhaps the better question to ask does not concern whether people get what they want, nor even how can we find out what people want, but the more fundamental question of how people can know what they want. We shall return to this last question after the next section, where we examine attempts to alleviate disappointment by the offering of a palliative (something that relieves but does not cure).

Palliatives

One school of thought advocates the use of standards as a way of solving I.S. problems. This line of reasoning assumes that the cause of the problem is that the methods used are not applied with sufficient rigor, so that misunderstandings and poor communication between customers and developers, and between developers, creep in. Try harder. However, this optimistic approach, that there exists the 'optimal' solution, and all you have to do is seek it more carefully, belies the fact that optimality is a non-existent state of affairs in business I.S. Furthermore, the rapidly evolving world of computer hardware, software, operating systems and environments, distributed and network developments, not to mention rapidly changing theories of management, organizational structures and behavior, and socio-economic theories, defeat a solution based almost entirely on standards. Standards for what?

Cousins of standards are evaluation and review. These concepts are also based on the notion that if you try hard enough, you will get it right. Review enables you to look back and change your mind periodically, so that system requirements can be refined into a correct version. Evaluation is designed to constantly check that objectives are being met, a means of reinforcement and remembering.

A very popular palliative is prototyping (Dearnley and Mayhew, 1983). Prototyping is a method of quick systems component construction, which is adopted within the framework of a broader system development approach or methodology to solve contentions between analysts, users, software and hardware. Such optimism is based on the idea that everything can be got right in the end if you go around the problem enough times. This approach has

advantages over the 'get it right first time' approach, but there is little evidence of any significant improvement in I.S. development.

Incremental development is based on the notion that you can creep towards where you want to be. This approach accepts that you cannot know what you want, well, not straight away. So far better to head off in the best direction by building part of the system, and then changing direction as necessary based on this experience for further development. This way you will get closer to where you want to be, wherever that is. The approach has the advantage that the customer(s) is taken along with it, and is therefore probably more agreeable with the system as it develops.

User Participation is another popular attempt to resolve the irresolvable (Land and Hirschheim, 1983). By getting user participation in decision making about the project, and/or in the actual analysis and design, the analyst is more likely to elicit the requirements of the system, and in the latter case, make it presentable to the user. End User Computing similarly attempts to involve the user, but now by handing over some of the system to the users to develop themselves, perhaps supported by I.S. professionals. The user is still in no better position to know 'what he wants', but is no position to complain about it, since it is his fault.

Some methods appear to confuse the difference between planning for the future and planning the future. The future is known by someone maybe, but how do you predict which person this is? All these palliatives attempt to take an existing way of thinking about I.S. development, and make it work by tinkering with it. It is the contention of this paper that the underlying way of thinking, or the paradigms used for system development, is themselves wrong for the reasons given in the next section.

Fixed Point Theorem

All of the discussion so far has been based on IS paradigms that have one common fallacy. This fallacy is expressed through the following 'mock' theorem.

The Fixed Point Theorem of Information Systems

There exists some point in time when everyone involved in the system knows what they want and agrees with everyone else.

Why is this theorem assumed to be true? Because we want to build the system to an exact specification.

Extension to the Fixed Point Theorem

The fixed point in the theorem remains fixed for the project duration, or even longer.

Implications of the Theorem and Extension

The implications are clear. Systems are built for one (hypothetical) point in time, whereas the system must work over some time continuum. Hence, disappointment. Building IS with the implicit, or sometimes even explicit assumption that the Fixed Point Theorem is true is guaranteed to cause problems. Before discussing ways of thinking that might not be based on the Fixed Point Theorem, it might be further illuminating to look at the reasons why this state of affairs has occurred.

How Did This Happen?

Grindley (1986) gives an excellent resume of how we arrived at the current state of the art in IS development in the chapter entitled 'The History of a Mistake'. One major cause is the project based nature of development. A project based approach is inherently finite time horizon driven whereas the environment is infinite horizon. This generates a number of differences between the desire to build something, and the need to make it work. These include targets versus standards, the inspirational motivation of a project team versus 'the pride in a job well done' motivation of the personnel running the system, the temporary society of the project developers versus the permanent society of the users, building versus maintaining. There is no doubt that a project mentality inherently begs for the Fixed Point Theorem to be true. This is exemplified by the project 'sign-off' where the customer agrees with documents of questionable interpretation, to take that version of the system as something that is 'acceptable'. Paul (1993) offers many more reasons for our current predicament.

Living Metaphors

So far we have discussed a major reason why IS disappoints. In this section, we shall look at possible ways of thinking (not solutions!) that might help. We have observed that the Fixed Point Theorem essentially leads us to the development of dead systems, systems designed to meet the needs of the business at one point in time. What is required for living businesses, however, are living systems. Systems should breathe, be designed to adapt to unknown change. Systems should grow, mature, and die (they do anyway!). Very few aspects, if any, of a businesses' operations are expected to be designed for life, static, so why should IS? A successful business is constantly adapting to change, and therefore so should the IS. The question of how this should be done must not be allowed to determine what it is that we want to do. Thinking of a living system development may be difficult, but without remembering that that is what is required, the Fixed Point Theorem will undoubtedly re-emerge consciously or unconsciously. The following discussion concerns some thoughts on living system development,

but is not meant to be an/the answer. We have a long way to go yet.

It is worth mentioning another major factor in a living system. This is the question of right and wrong, the question of correct and faulty. A living system is neither right nor correct. It is a system that is adapting to need. In this sense, it is not wrong or faulty either. The expectation should be that the system is constantly undergoing change to meet the changing need. The opportunity to change the system has to be seen as a virtue in itself, not a point of negative criticism concerning a past lack of perspicacity. This is not to say that mistakes cannot be made, since individuals will continue to do that, but criticism of the system encourages cover-ups, a get it right mentality, and back to a dead system.

The Gardening Metaphor

Gardens constantly change, evolve, and grow. A basic architecture may be laid down - paths, trees, and lawns. But the smaller plants come and go, can be moved to some extent, and everything is evolving with the seasons. Gardens require incessant attention - pruning, weeding, and reduction of bugs and plant predators. Excessive action leads to temporary empty spaces, or islands of apparent inactivity. Lack of attention can lead to choking, overgrowth, and the death of some plants at the expense of others.

The gardening analogy fits the needs of business systems very well. It also exemplifies the earlier point about success and failure. The latter can be quite obvious. But is a garden ever finished, complete, or successful (except at the Chelsea Flower Show!)? A garden may be adequate to please visitors, but the gardener is never satisfied. There is always something to improve upon, something to tend to as the rest of the garden moves on. In a dynamic, changing, uncertain world, this is what is required from a business information system as well.

The Centre for Living Information Systems Thinking (LIST) at Brunel University

Paul and Macredie (2000) present a selection of contributions to LIST which cover 5 case based studies in the power industry, retail sector, software development, health service, and logistical distribution. The methods examined include business fit, tailorable IS and deferred system's design, component-based software architectures, stakeholder analysis, and business process re-engineering. Three further papers make theoretical contributions based on architectonics, LIST sustainable environments, and task analysis for IS evolution.

Conclusions and the Change Paradox

First, living system development is a way of thinking, not a solution methodology. The latter leads straight back to the Fixed Point Theorem. Second, demanding 'success' will lead to failure (because to know you have success requires that you know what you want, i.e. the Fixed Point Theorem). Third, in spite of the immense cultural and economic pressures, the project based approach has to be broken. Projects have time and budgetary deadlines that feed off a requirements declaration, for example in software engineering. Software engineering should not be compared to any other engineering. It may be time to move on and away from Software Engineering altogether.

We conclude this paper with a paradox concerning the contribution of the IS industry to it's customers' solutions – or is it problems? Successful I.S. development of a living systems kind will enable a business to compete more effectively in its market place. This will in its turn put more pressure on the competitors to be more competitive else they will lose market. If they turn to I.S. development to successfully facilitate competitive advantage, this will in its turn ratchet up the competitive pressure in the market. Table 1 illustrates this change paradox, whereby successful I.S. leads to pressure for more change.

Table 1: The Change Paradox

Statement	**Observation**
Companies are changing to meet competition	I.S. might assist
If company rate of change much much less than IS development speed	OK
If company rate of change a bit less than IS development speed	Familiar problems
If company rate of change a bit more than IS development speed	Failures
If company rate of change much much more than IS development speed	Impossible
IS enables/facilitates competitive advantage	Company change can speed up
Company change speeds up	Go to top of table

References

Dearnley, P.A., and Mayhew, P.J. "In Favour of System Prototypes and Their Integration Into the Systems Development Cycle" *The Computer Journal* (26:1), 1983, pp.36-42.

Grindley, K. *The Report, Fourth Generation Languages: Volume 1, A Survey of Best Practice*, IDPM Publications Ltd, London, 1986.

Land, F., and Hirschheim, R. "Participative Systems Design: Rationale, Tools and Techniques," *Journal of Applied Systems Analysis* (10) 1983,pp. 91-107.

Paul R J. "Why Users Cannot 'Get What They Want' ". First published in *The CRICT Workshop Papers*, J. Low and S. Woolgar (eds.) CRICT, Brunel University, UK, 1993, pp. 1-10.
Reprinted in:
ACM SIGIOS Bulletin, (14:2), 1993, pp 8-12.
New Zealand Computer Society Quickface Newsletter, 1994. pp. 1, 14-19.
International Journal of Manufacturing System Design, (1:4), 1994, pp. 389-394.

Paul, R. J., and Macredie R.D. (eds.) *Living Information Systems Thinking Towards Organisationally Relevant Information Systems*, to be published by Springer, London, 2000.

Watts, S. "Of Mammon and the Machines He Employs," *The Independent On Sunday*, 8 November, London. 1992.