27 Issues

27.1 What is a methodology?

In Chapter 2 we provided a working definition of the term 'methodology', and, although this has been adequate for our purposes, we will now look in more depth at the question 'what is a methodology?'. The term is not well defined either in the literature or by practitioners. There is very little agreement as to what it means other than at a very general level. The term is used very loosely but also very extensively.

This loose use of the term does not, of course, mean that there are no definitions, simply that there are no universally agreed definitions. At the general level, a methodology is regarded as a recommended series of steps and procedures to be followed in the course of developing an information system. In a brief *ad hoc* survey, this proves to be about the maximum that people will agree to, and, of course, such a definition raises many more questions than it answers. For example:

- What is the difference between a methodology and a method?
- Does a methodology include a specification of the techniques and tools which are to be used?
- Does a collection of techniques and tools constitute a methodology?
- Should the use of a methodology produce the same results each time?

The questions that arise are fundamental as well as numerous. Unfortunately the problem will not be solved here; the most that can be achieved is that the issues will be aired. The information systems community has regularly debated the meaning of the term methodology in an information systems context, and as yet it has not come up with any universal definition.

One of the most useful definitions for the authors is that provided by the British Computer Society (BCS) Information Systems Analysis and Design Working Group as long ago as 1983. They defined an information systems methodology as:

> a recommended collection of philosophies, phases, procedures, rules, techniques, tools, documentation, management, and training for developers of information systems.
>
> (Maddison, 1983)

Utilizing this definition, we suggest that a methodology has a number of components which specify:

- what tasks are to be carried out at each stage;
- what outputs are to be produced;
- when, and under what circumstances, they are to be carried out;
- what constraints are to be applied;
- which people should be involved;
- how the project should be managed and controlled;
- what support tools may be utilized.

In addition, a methodology should specify the training needs of the users of the methodology. On the other hand, the frameworks discussed in Chapter 26 are far less prescriptive.

Apart from the above, we believe that a methodology should also specifically address the critical issue of 'philosophy'. We mean by this the underlying theories and assumptions that the authors of the methodology believe in and that have shaped the development of the methodology. This identifies those sometimes unwritten aspects and beliefs that make a methodology an effective approach to the development of information systems in the eyes of their authors. We believe that the definition of a methodology should include specific reference to its philosophy as this has a critical bearing on the understanding of a particular methodology. An information systems development methodology is therefore much more than just a series of techniques along with the use of software tools.

Utilizing these ideas and beliefs we extend the BCS definition of a **methodology** as follows:

> A systems development methodology is a recommended means to achieve the development, or part of the development, of information systems based on a set of rationales and an underlying philosophy that supports, justifies and makes coherent such a recommendation for a particular context. The recommended means usually includes the identification of phases, procedures, tasks, rules, techniques, guidelines, documentation and tools. They might also include recommendations concerning the management and organization of the approach and the identification and training of the participants.

Some methodologies are, of course, more comprehensive than others, which is why we include the statement about 'development or part development' and, as mentioned above, the 'rationales and underlying philosophy' are not always made clear but we believe that they always exist and are the key to understanding a particular methodology. For example, they will shed light on the ethical stance that a methodology incorporates. We include 'a particular context' in the definition because it is clear that methodologies are not universal in their applicability, despite some methodology authors seemingly thinking they are. For us there are always limitations to the applicability, and thus we believe that those contexts and limits should be addressed. Together the philosophy and the context help illuminate the assumptions that the authors of the methodology are making, which is why we do not specifically include a reference to 'assumptions' in our definition. In practice, many methodologies, particularly commercial ones, are products that are 'packaged' and might include:

- manuals;
- education and training (including videos);
- consultancy support;
- tools and toolsets;
- *pro forma* documents;
- model-building templates, and so on.

Some have argued (e.g. Flynn, 1992) that the term 'methodology' is not apt in the context of systems development and that the term 'method' is perfectly adequate to cover everything that we mean by a methodology. Indeed Flynn states that 'the term methodology was popular for a time in the 1980s' implying that it is no longer much used. This is contrary to our experience, although it is true that the term 'method' is also used. For us, this might be argued to substitute one ill-defined word for, another, but more importantly we believe that the term 'methodology' has certain characteristics that are not implied by 'method', for example, the inclusion of 'philosophy' which we have already suggested is key. Methodology is thus a wider concept than method.

Checkland (1981) has distinguished between the two terms and says that a methodology:

> . . . is a set of principles of method, which in any particular situation has to be reduced to a method uniquely suited to that particular situation.

In Checkland (1985), he argues that information systems development must be seen as a form of enquiry in the context of the general model of organized enquiry, which consists of three components: an intellectual framework, a methodology, and an application area. This suggests a hierarchy of elements to enquiry.

The first element is the **intellectual framework**, which comprises the ideas that we use to make sense of the world. This is described as the philosophy that guides and constrains the enquiry. It consists of ontological assumptions, that is, beliefs about the fundamental nature of the physical and social world and the way it operates, and epistemological assumptions, that is, the theory of the method or grounds of knowledge. These terms are discussed in Section 28.2. It also consists of ethical values, which should be articulated, that may serve to guide or constrain the enquiry.

The second element is the methodology. This is the operationalization of the intellectual framework of ideas into a set of prescriptions or guidelines for investigation which require, or recognize as valid, particular methods and techniques.

The third element is the application area; that is, some part of the real world that is deemed to be problematical and worthy of investigation.

This is a useful framework for discussions of research and enquiry, but we need to know how well it relates to the world of information systems development methodologies. Working backward, it would seem that the application area is that of information systems development in general. The methodology element is equivalent to the collection of phases, procedures, rules, methods, and techniques that are usually considered to be a methodology in the information systems world. The intellectual framework is the element that is often missing from methodologies, or rather it is not missing, for it exists, but is not explicitly articulated. In our definition, much of this intellectual framework element is included in what we term the underlying philosophy, which we include in the definition of methodology itself.

27.2 Rationale for a methodology

It is important to discover why people adopt a particular methodology. Obviously this varies substantially between organizations and individuals, but we can identify three main categories of rationale: a better end product, a better development process, and a standardized process.

1 A better end product

People may want a methodology to improve the end product of the development process, that is, they want better information systems. This should not be confused with the quality of the development process, addressed below. It is difficult to assess the quality of information systems produced as a result of using a particular methodology. For example, we cannot know that the use of the methodology produced the particular results. The same results might have occurred if the system had been developed using another methodology or without using any methodology at all.

Even if we had some way of comparing the results of using different methodologies, the elements that are perceived to constitute measures of quality differ considerably from person to person, and there is little agreement within the information systems community on this issue. The following represents our attempt to address some of the components of quality of an information system:

- *Acceptability* – whether the people who are using the system find the system satisfactory and whether it fulfils their information needs. This includes business users and managers and their requirements.
- *Availability* – whether it is accessible, when and where required.
- *Cohesiveness* – whether there is interaction between components (subsystems) so that there is overall integration of both information systems and associated manual and business systems.
- *Compatibility* – whether the system fits with other systems and other parts of the organization.
- *Documentation* – whether there is good documentation to help communications between operators, users, developers, and managers.
- *Ease of learning* – whether the learning curve for new users is short and intuitive.
- *Economy* – whether the system is cost-effective and within resources and constraints.
- *Effectiveness* – whether the system performs and operates in the best possible manner to meet its overall business or organizational objectives.