The use of Z

Rosalind Barden, Susan Stepney and David Cooper

Logica Cambridge Ltd

Abstract

During 1990 and 1991 we carried out a survey of Z users in the UK; in this paper we present the results. Z is being used by a wide variety of companies for many different applications. Many institutions use Z because they choose to, rather than because it is mandated by a defence or security related client. Half of the participants in the survey were first-time users of Z; with a little training and some expert guidance they were soon able to produce Z specifications. Nearly all the institutions who have used Z intend to do so again; those who don't plan to use a formal method more appropriate to their needs. We did not uncover many really large projects using Z, but most of those which we did survey are of a reasonable size. Over half the projects surveyed are using tools, with the majority of them employing type checking support. Tools use grows with the size of the project, although several large specifications have been produced without the use of tools. Not many people are proving their specifications, nor stating desirable theorems or proof obligations. However, many people expressed an interest in this with the unavailability of appropriate tool support being given as a reason for not attempting proof.

1. Introduction

ZIP is a three year project concerned with enhancing the use of Z. As part of this work we conducted a survey of Z users. The aims of the survey were to discover: how Z is being used; to what types of project it is being applied; what sort of help and advice Z practitioners need.

The survey was carried out in two phases: the first in the summer of 1990, among the ZIP partner organisations; the second during the spring and summer of 1991, among the wider Z community.

In conducting this survey we have relied upon volunteers coming forward. We have tried to obtain a range of people to survey and have sought contact with particular organisations which we felt to be prominent or addressing an application area which we would otherwise have missed. This survey should not be regarded as a scientific study since the sample was in no way controlled. Nonetheless, we feel the results presented here will make interesting reading and help to provide some information about the use of Z today.

The survey was based around a two part questionnaire. It was carried out mainly by face-to-face interviews conducted at participants' sites; some participants completed their questionnaires by post.

The first part of the questionnaire sought factual information about an organisation's use of Z and covered:

- previous use of formal and structured methods
- experience of staff involved in specification and implementation
- reasons for using Z
- size of specification and effort required to write it
- features of the Z notation used

Rosalind Barden, Susan Stepney, and David Cooper. "The Use of Z."
In J. E. Nicholls, editor, Proceedings of the 6th Z User Meeting, York 1991, Workshops in Computing, 99-124. Springer-Verlag, 1992

The second part of the questionnaire consisted of discussion questions which formed the basis of the interview. These questions sought the opinions and views of the interviewee about the way Z had been used. They addressed:

- style of Z used and any method adopted
- comparisons between Z and other formal notations and structured methods
- iterations gone through
- use of proof
- how validation occurred
- tools used
- experience and reflections on the project
- recommendations for first time users

In total 38 face-to-face interviews were performed and 8 postal responses received, giving a total of 46 responses. Some people provided details of more than one project, giving 56 projects in total. The statistics presented in this paper are mainly on a per person basis, except where it is relevant to calculate the figures on a per project basis. The survey covered the major commercial users of Z in the UK, together with some members of academic institutions.

A list of participants is given in appendix A.

The full report of this survey forms a deliverable of the ZIP project [Barden *et al*, 1991]. It contains the details as presented here, along with a full analysis of the use of the Z notation, a summary of some Z projects as presented in the literature, and a report on what interviewees would like to see in a Z methods handbook.

2. Experience of the users

As might be expected with a fairly new technique, about half of our subjects were newcomers to Z. Many of the projects were performed by people who had little experience of the notation; usually what knowledge they had been acquired on a short industrial course (figure1).

short course	65%
case studies	18%
no preparation	18%

Figure 1. Where first time Z users gained experience

However, many people had previous experience or knowledge of other formal notations and structured methods. Figure 2 records how much experience the interviewees had and how much the team with which they were working knew. (Note: some of the projects had only one person in the team and so the information about the team will have been irrelevant.)

formal notations		structured methods		Z		
you	team	you	team	no. previous projects	you	team
72%	72%	57%	59%	0	48%	56%
				1	18%	8%
				2	15%	4%
				3	10%	4%
				4 or 5	3%	12%
				6+	8%	16%

Figure 2. Formal and structured methods experience of Z users

Just about everyone involved had a numerate background, which probably reflects the composition of the industry as much as anything, but this did not always mean that they were familiar with the discrete mathematics required for Z.

3. The specification document

3.1 Specification statistics

Although we surveyed a few very small projects, notably in the academic institutions, most were of a reasonable size. On average the projects took 134 staff-days of effort. The average length of specification was 143 pages. This gives us some confidence that Z is being applied to genuine problems. We did not discover many projects with large teams of people (figure 3), so it seems that Z has yet to become common among very large projects. We have examined the breakdown of team size into readers and writers, but can see no common pattern.

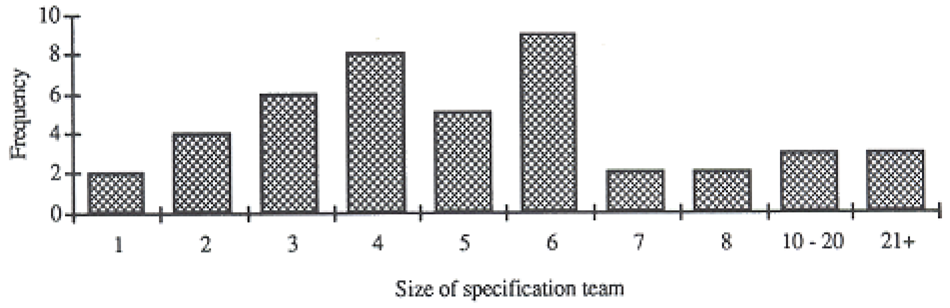

Figure 3. Size of specification teams

Several respondents expressed concern about whether Z would scale up to very large projects, for example, 400 people working in 4 different countries. Tool support was seen as necessary for this type of project, yet generally tools have been developed for stand-alone terminals, and do not work adequately with large specifications, on large, distributed, multi-user environments.

The sizes of the schemas, and other chunks of Z, varied quite considerably between projects (figure 4), but judging by the few examples we saw, the size of schemas appears to be reasonably steady across any given specification. So it seems that the variation in size is caused by local styles.

1 – 5	6 – 10	11 – 20
18%	52%	30%

Figure 4. Number of lines per schema

44% of the respondents claimed that their specifications had the same amount of mathematics as English (figure 5). Of those with more mathematics than English, there was an appreciation that they should be striving for a 1:1 ratio. Several people commented that the use of Z had led to a better English description.

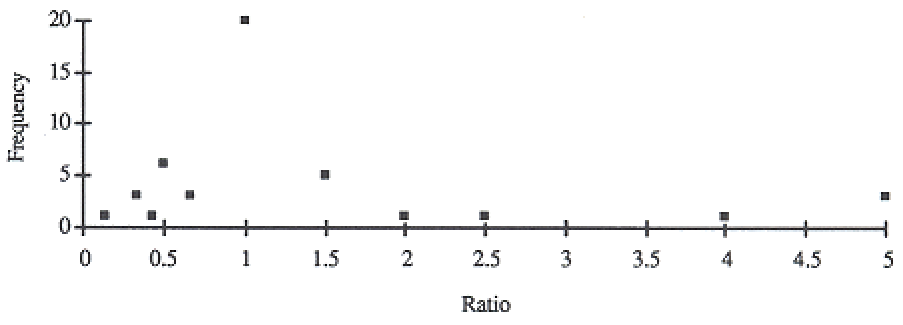

Figure 5. Ratio of mathematics to English in Z specifications

The average rate of writing was 1.6 pages/staff-day, with a standard deviation of 0.9. Figure 6 is a scattergram showing how the number of pages compares with the number of days effort required to write the specification.

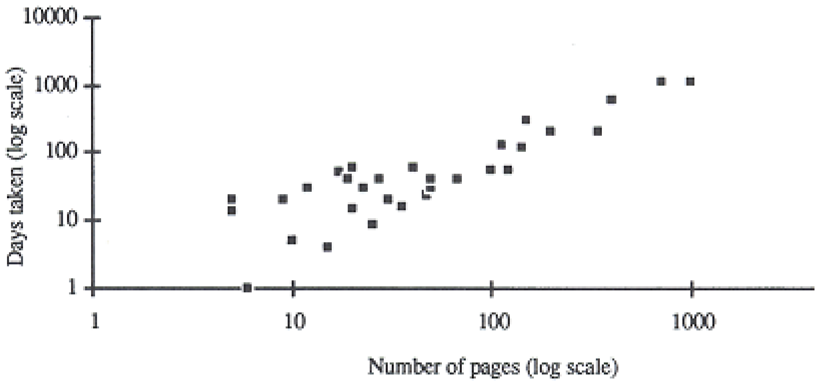

Figure 6. Ratio of mathematics to English in Z specifications

We have performed a comparison of the daily page rate with the ratio of mathematics to English and can see no relation. However, as figure 5 shows, the number of projects for some ratios is quite small: it would be unwise to draw any conclusion from this analysis.

3.2 Specification style

Nearly everyone was using the state and operations approach promulgated by the Oxford University Programming Research Group. Four projects were using a functional style (an example of such a style is [Mitchell *et al*, 1991]) and several people expressed the view that they intended to move to a functional style in the future. A couple of people had tried an object oriented approach (a survey of object oriented Z approaches can be found in [Stepney *et al*, 1991], [Stepney *et al*, 1991]); one of these used an object oriented approach to drive the specification and within this used state and operations. Some people had been influenced by their previous experience of structured methods in the way that they wrote their specification, and at least one project used an SSADM style to fit in with the rest of the specification and the client's needs.

3.3 Iterations

When asked how many iterations had been undergone in order to produce the final specification, many people had difficulty giving an exact answer because of the nature of software specification, which tends to proceed without too many landmarks. However, those who did follow a more formal review process were able to give us a figure for the number of iterations (figure 7). Nearly all of the projects undertook two or three iterations, which often meant that the first one was carried out before the full functionality was in place.

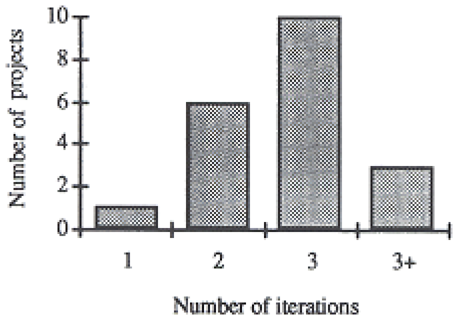

Figure 7. Number of iterations of a specification

The typical result of the review process was that the amount of English in the specification would increase, since more explanation was deemed necessary, whereas the amount of mathematics would decrease, as better and more abstract ways were spotted for modelling the problem.

3.4 Standards or Styles for Z

Not many people had an in-house standard or style for the use of Z, although many said they would like one or were thinking of developing one. Five projects had operated under a strict standard, and about the same number again had local rules such as naming conventions and standard approaches. A common opinion was that it was too early to develop a standard style for the use of Z. Many organisations had only recently started to use Z (recall that 48% of respondents were first time users) and were not yet ready to commit to a standard approach. This is further borne out by the fact that where standards did exist, they were within organisations which had been using Z for some time.

Of those who did have a standard, this was often to keep in line with the tool they were using. One organisation had developed an approach which determined how the documents should be structured and what components should be included. This work had been undertaken for a client and was not yet adopted company wide.

People have quite varied expectations of what might be contained in an in-house standard, for example, naming conventions, syntax defined in a particular tool, methodical approach, and document structuring.

3.5 Tool support

During the period of the survey, more tools became available and a number of companies developed tools for their own use. (For a catalogue of Z tools, see [Parker 1991]). As the survey progressed we found that the use of tools became more common. Overall, 58% of the projects used tools.

When asked what they would do differently next time, a number of the people who had not used tools said that they would in future. Nonetheless our findings do suggest that it is practical to write Z specifications, at least of the size which we surveyed, without the use of tools. However, the use of tools did increase with the size of the project (figure 8).

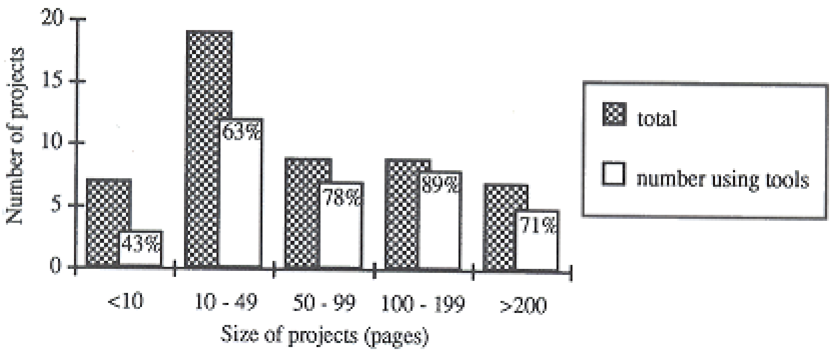

Figure 8. Use of tools compared with size of project

Note: not every project gave us information about the size of the specification produced; the numbers in figure 8 include only those projects for which we did know the size of the specification document.

36% of the projects which used tools used them only for font support and general production of the specification. 64% were using tools which offer type checking and support for building the specification, such as pro formas, browsing and schema expanders. There was wide recognition, even among non tool users, that type checking was valuable for saving time, since it enabled silly errors to be trapped. Indeed those that had undertaken larger projects said that a tool was essential for such work. In addition four projects had experimented with or used proof tools, and a further project was hoping to acquire such a tool shortly.

Several people commented that tools can be too restrictive. When a specification is under development details are often left until later to be filled in; tools usually do not support this type of development. As one interviewee put it 'It is important to be able to bypass formality when appropriate. One should not be a total purist'.

3.6 Validation

In general, we did not find that projects were using any special techniques to validate their Z specifications against the informal requirements. The same review processes were applied to the Z specification as might have been used for any other type of specification; for larger companies this process was dictated by in-house quality standards.

A few projects had tried animating their specifications and Prolog was regarded as useful in this respect. In at least one project animation had revealed errors in the specification. A direct translation from Z to Prolog is fairly straightforward, but performance is poor [Dick *et al*, 1990]. If animation is to gain more widespread use, then better support for it will need to be developed.

3.7 Proof

Proof activity is growing, although we surveyed only one project which acknowledged that proof had played a major part of the work. Quite a few respondents are interested in finding out how to conduct proof, particularly how to identify sensible things to prove. Several people are investigating the acquisition of proof tools.

The proof work varied from simply stating a few obligations, to proving theorems about the system formally. Figure 9 shows the number of projects surveyed undertaking some form of proof. (Note: projects are included under one category only, the ones proving theorems are also recording them!)

record obligations	4
discharge obligations	7
state theorems	1
prove theorems	6
prove refinement theorems	1

Figure 9. Number of projects performing aspects of proof

Proof activity currently is at a low level; projects have been included in figure 9 if they carried out *any* work in one of the categories, even if they proved only *one* theorem. Nearly all the work is concerned with demonstrating desirable properties of the specification. As a further indication of the amount of proof work being undertaken, when asked whether they used the precondition operator **pre** *S*, a mere six interviewees replied that they used it habitually.

Several respondents mentioned that the lack of tool support for proof was a limiting factor. Interviewees complained that hand proofs are tedious; Z specifiers need mechanical assistance with checking proof work and in proving simple lemmas.

3.8 Implementation

Not all of the projects which were surveyed went on to be implemented. Some of the projects were a post hoc specification of an existing system; some did implement but not from the Z specification; some were written in order to investigate some properties or to specify grammars. Of those which were implemented from the Z specification, most of the team sizes were small (figure 10). 78% reported that they had enjoyed coding from the Z specification, one project said that it made no difference.

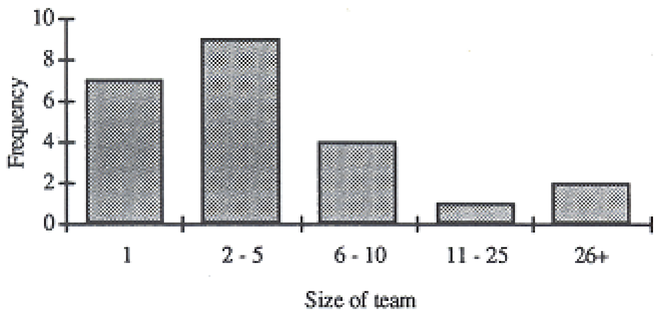

Figure 10. Frequency of size of implementation teams

4. Why use Z?

4.1 Reasons for choosing Z

Interviewees were asked why they had chosen Z (figure 11). More than one option could be given, and quite often was. Unsurprisingly for a new notation, many people were trying it out because they wanted to, or for evaluation purposes. Two projects were evaluating Z for a client. Sometimes clients required the use of some formal method and the project had made the decision to use Z.

Personal preference	49%
Evaluation of Z	38%
In-house requirement	22%
Client requirement	31%
Other	16%

Figure 11. Reasons for choosing Z for the project (more than one could be given)

Other reasons given included ‘in order to promote the use of Z within an organisation’ and ‘to complete an MSc project’.

4.2 Type of projects on which Z has been used

We do not have a clear view of the type of projects for which Z is suited, and for which it is not suited. However, figure 12 shows the projects for which Z has been used, and all but two of the respondents indicated that they would use Z again. Of the two who will not use Z again, this is because they are now using a formal notation more suitable for their needs. A further two respondents were not sure — one would wait to see if a suitable application came along; the other would consider using another (possibly formal) notation more suitable for requirements specification.

security related	29%
data/text processing	17%
Z research/student project	10%
tools	10%
algorithm design	8%
safety-critical	8%
hardware	6%
defining grammar or standard	6%
operating system design	4%

Figure 12. Type of project on which Z used

Notes on these categories:

1. Clearly a project could potentially be in more than category. We have put each project into one category only, using the widest description possible, for example, a secure operating system would be classified as security related.
2. The category of Z research includes research projects being conducted in commercial institutions.

It is often said that people will use Z only where they are required to, hence only on a defence or security project. This is not true. There are notable published exceptions: [Barrett 1989], [Brownbridge 1990], [Garlan & Delisle 1990], [Hall 1990], [May 1990], [Nash 1990], [Nix & Collins 1988], [Phillips 1990], [Shepherd & Wilson 1989]. Our survey has found further projects which do not fall into the defence or security category. However, of the projects for which the client required the use of Z, three quarters of them were security related.

5. Case Histories

To give a more ‘living’ picture of the way that Z is being used than the raw statistics given so far, we present here a description of the way five of the companies surveyed used Z. We have kept the names of the companies involved confidential.

5.1 Company A

We interviewed several team members at company A, including the project manager. This manager has a strong interest in how formal methods work influence managers, is very positive about the use of formal methods, and points to the increase in confidence and predictability of the projects in which they are used. The use of Z has resulted in improved use of senior staff, who, after writing specifications embodying their skill and experience, could pass on their work to less experienced people.

Z was used on the project as a result of personal preference and also as an evaluation exercise. The project team consisted of 20 writers. 40 people were involved in reading the specification, although some of these were also writers. Most of the team enjoyed using Z, although a few, who could not see the point, found it a trying experience. All team members received training in Z before starting work on the project. Those who had a grounding in mathematics did find it easier.

The specification was large and was written in a state-based style. Company A approached the specification by first defining the basic state, then the operations. While writing the specifications, there was lots of thinking and consultations with Z experts. One part of the system specification went through two iterations. Company A used formal reviews and walkthroughs to validate their specification. Sometimes the specification changed as a result of better structuring.

The system proceeded to implementation with some refinement being performed. The refinement process was less successful than the use of Z for specification, and although company A will use Z again, if a better method of refinement came along they would try that.

An important lesson that this company draw from their experience is that it is vital to spend time at the beginning ensuring that the Z specification is correct before embarking on implementation. Although this means that the start of coding will be delayed, once started the implementation will proceed quite rapidly.

Company A are very pleased with their use of Z and the feedback from their users on code quality has been excellent.

5.2 Company B

The leader of the project surveyed at company B was an experienced user of Z, having had several years of Z experience and having used Z on six previous projects. The project team consisted of the experienced writer and five other members who were mainly involved in reading the specification.

The specification was concerned with a safety critical control system. It was performed as an evaluation exercise for a client, to see how well Z could be used in their company.

The client was not familiar with Z, so a simple subset of the language was used. Additionally a ‘straight maths’ form was included as an appendix, since the client was more familiar with this. Some of the messy detail was specified only in this appendix, which meant that the Z part was more abstract; this resulted in simpler proofs.

As part of this project a natural language description of the problem and a proposed algorithm were both specified in Z, and then correctness proofs were attempted. This exercise showed that the algorithm did not in fact solve the problem.

The method adopted on this project was to start by writing down everything and then remove the parts which were not needed. This process is not linear; sometimes bits had to be put back in again as the writer discovered that they were needed after all! The process continued until as high a level of abstraction as possible has been reached. Developing the specification went through many iterations; these made the specification more abstract and corrected errors. In the end the specification was about half the size of the original. The rate of production was about 1.3 pages per day.

The level of abstraction required varies according to the purpose of the specification. In this case it was for capturing requirements so a high level was beneficial. As part of another project carried out by company B, the aim was to evaluate an informal design and the level needed had to correspond to the abstraction level of the given design, which was much lower.

The specification was validated by proving properties and by formal walkthroughs. As a result of seeing the formal specification, the originators of the case study ended up by arguing amongst themselves about what the problem really was!

Company B's client was very pleased with the results of this evaluation exercise.

Company B made a plea for the Z proof rules to be sorted out, and for Z to be standardised, so that it meets the requirements of DefStan 00-55.

5.3 Company C

The aim of the project at company C was to understand the needs of security in databases, and in particular on how additional information can be inferred from the information extracted from the database: can classified information be inferred from sufficient unclassified output?

The specification consisted of a database and a definition of what it means for the database to be secure with respect to inference. Company C carried out proof work to show that a set of more implementation oriented constraints were sufficient to guarantee security.

Prior to this project much of the work at company C had been in another formal notation which is better supported by tools. Company C have experimented with a number of Z tools, ranging from font support to type checking. They will use Z again, but will use other notations too if these are appropriate.

A very iterative approach was used and many versions had to be thrown away before an acceptable model was found. The specification started at a high level as this helped the team to concentrate their ideas on the important parts and not be overcome with details. The effect of the iterations was to increase the size of the specification as more parts were added that had not previously been considered. The existing parts reduced in size as the specification was made neater as the work progressed.

There was much proof activity on this project — about 40 pages of proof accompanying around 50 pages of specification. During the proving process the specification was changed to make the proofs easier. Company C recognise that some of these changes may have been due to errors in the original specification. It was generally found to be better to reduce statements to a 'simplest' form, which usually meant using more quantifiers instead of fancy symbols or function arrows. It was easier to prove things if all the predicates were written out fully and simply.

An important lesson which company C would wish to pass on is that formal methods do not make difficult problems go away: a messy problem will tend to lead to a messy specification.

The specification writers both had some Z experience, one acted as the main specifier while the other performed most of the proof. They would recommend any beginner to discuss their work with an experienced Z user.

5.4 Company D

Company D has previously used Z on a number of projects in the military and security arenas. The project surveyed was concerned with a video interface to a collection of databases holding a wide range of information. The application was purely civilian and the decision to use Z was made by the team members. They wished to evaluate the use of Z for such work.

The team members had knowledge of other formal notations and of many structured notations. One team member had attended a Z course, and the team were able to benefit from the advice of a more experienced Z user from another part of the company.

The specification done for this project was an abstract one; because of time constraints and the relative inexperience of the team members in Z, the part of the specification which addressed exception handling was not covered.

The bulk of the specification work was performed in SSADM version 3 which was known to the team and to the client. An SSADM style was used to drive the Z specification and the work performed in Z was equivalent to SSADM stage 5 (process design). Given their experience, the team feel that it is important that the links between notations such as SSADM and Z be developed further, especially where the target language is a 4GL.

The Z specification did not form part of the deliverable design documentation, but was included in the documentation set given to the client. The consultants who performed the technical review of the project were able to follow the specification.

Using Z meant that the specification took a little longer to write than it might otherwise have done. However, the use of Z provided a number of advantages:

- The implementation was performed directly from the specification. The common mathematical basis for Z and the target language made it a natural specification method to use.
- It allowed the algorithms to be designed far more easily than if Z had not been used.
- Z made it easier to stipulate the complex behaviour of some of the functions involved in the database retrieval, and assisted in spotting invariants in the data.

As there was no tool support available, type checking of the specification was performed by hand by staff from another part of the company who had more experience in using Z. This worked for this fairly small project, but the team know that it would have been better to have used a tool.

The team members enjoyed using Z and this project and found it beneficial. They are currently looking for further opportunities to use Z in their work.

5.5 Company E

The team at company E had used Z on a previous project, and the company had used Z and other formal notations on several projects. The project involved a team of one person writing the specification and two people reading it. The writer had extensive knowledge of Z, having had five years of Z experience prior to this work.

The specification produced was over 100 pages long, and took about three months to write. The specification was written in the state and operations style, which is an in-house optional standard.

Company E tried to have only one level of specification, because of document control overheads and consistency problems. This meant that the specification had to be of a sufficiently high level to be concise, yet at a low enough level to support implementation. In the end they found that they needed two different specifications, for their two types of users:

1. an interface specification, at an abstract level, to be used by clients
2. an implementation specification, at a lower level, to be used by maintainers

The people we interviewed at company E recommend that specifiers concentrate on obtaining the right state first of all, for then the operations will fall out almost automatically. If the operations are tricky to specify, this is an indication that there are problems with the state. Iteration will be needed to iron out such difficulties.

In general the team found that they needed three attempts at the state, but the operations were easier to do and only minor changes were needed for them. Other changes were made to the specification when generalities were recognised.

Company E mandates the use of formal reviews. During the reviews the specification tended to shrink, because it was decided that certain aspects were not worth specifying:

1. construction of character strings
2. components which were bought in

The team also developed some of their own shorthand which allowed them to say what they wanted quickly, if not in legal Z!

The project used a type checking tool. No proof obligations were discharged as it was considered to be too time consuming and unwarranted for this particular application. On other projects company E has used labelled steps and justifications to address proof obligations.

The specification was implemented in C directly from the Z, by a team of two who had not been involved in the specification. One of the implementors had two years of Z experience, while the other was a novice. They enjoyed working from the formal specification, finding it easier to work from a formal specification than their previous experience. The project team felt it would be useful if short cuts between the Z and C were documented. For example, two sequences in Z are the same if they have the same elements in the same order; in C they are the same if they have the same pointer.

An important lesson which company E draw from this project is that effort is required to keep formal specifications, formal designs and code consistent with each other.

6. Experience gained from the specification

We all have something to learn from the experience of others. We asked the survey participants directly about their recommendations for first time users. The most common suggestion was attending a course, perhaps reflecting the large number of responders who had themselves been on a course. Training was seen as essential, with remarks like 'the key element of training is practical content', 'it is essential to go on a course', 'users should go on a training course, it is not enough to have just one person who knows Z properly'. There was also an appreciation of the need for self training; a good starting point is to see what others have done by looking at relevant books and papers. One person recommended that newcomers should first read up on basic set theory.

Several people recommended a pilot project, although this was tempered with remarks such as 'it is important to have a follow-on plan'. One participant strongly recommended this approach, while another dubbed it 'ideal, but idealistic'. Having experienced Z users available was seen as important; this could be achieved either by recruiting experienced staff or by having a consultant available. Comments here included 'recruit experienced staff — it is vital to have an experienced leader' and 'get in touch with someone who knows about formal methods to use as a mentor and consultant'. A few people disagreed with the idea of hiring experts, preferring to train their existing staff. A particular comment which sums up the views of many is 'it is useful to have a guru around and also a project manager who is enthusiastic and committed'. The top four recommendations are given in figure 13 along with the number of interviewees giving each piece of advice.

attend course	22
read books and case studies	18
ensure expertise available	16
do a pilot project	12

Figure 13. Top four recommendations for first time users

The next most popular recommendation was to check that the project was ‘suitable’; unfortunately, little guidance is available as to what this means. However, we did gain some indications of areas in which it is difficult to apply Z (see below).

6.1 Difficulties with Z

Interviewees were asked about what is difficult or impossible to say in Z. Aspects of timing and concurrency were mentioned the most often. However, there was an appreciation that Z wasn’t intended to do everything. Remarks here included ‘cannot do concurrency in Z, but then it isn’t intended for that’ and ‘some things are impossible to say in Z — we knew this and didn’t try to say them’. Another common complaint was the difficulty of describing aspects which would be easy to do in an algorithmic language such as ‘do something for every member of this set until some condition holds’. Several people complained that the support for structuring of specifications offered by the schema calculus was insufficient.

Figure 14 records the top five most difficult aspects of using Z mentioned by people in the survey and the number of respondents who made each complaint.

timing aspects	9
algorithmic aspects, programming constructs	7
particular language features	6
sequencing operations	4
things to do with schemas	4

Figure 14. Top five most common difficulties with Z

As to areas which people found hard to use, recursion, preconditions, framing and the type system were mentioned. Theta was loved by some and loathed by others; those who dislike it avoid its use altogether, while its fans use it habitually.

Most users employ a subset of the notation. The most common reason given for not using any part of the notation was that, although felt to be a useful feature in general, it was not needed for that particular specification. The next most common reasons were that the specification was clearer without it, and that although not used the notation ‘had potential’.

Interviewees were asked to state whether they had used a particular part of the notation habitually, infrequently or never. In order to discover a ‘popularity rating’ for the notation, we subtracted the number of ‘never’ answers from the number of ‘habitually’ answers. Figure 15 presents the top and bottom fives from this ranking.

Popular notation			Unpopular notation		
schema definitions		+43	relational iteration	R^k	–24
schema inclusion		+40	total surjection	↠	–24
Delta convention	ΔS	+38	schema iff	$S \Leftrightarrow T$	–24
input convention	$x?$	+36	finite injection	⤕	–28
partial function	⇸	+34	bag notation		–30

Figure 15. Top and bottom five aspects of the Z notation

The function arrows were particularly unpopular; only the partial and total functions, and the partial injections obtained positive ratings. Most of the others were denounced as confusing, with many people rejecting them in order to make their specification clearer (nonetheless slightly more people still thought that the notation was useful).

In the case of the schema calculus only conjunction and disjunction gained positive ratings. A number of people drew attention to problems with schema implication and schema negation. The use of these two was explicitly prohibited on some projects. Figure 16 gives the popularity ratings for the schema operators.

conjunction	$S \wedge T$	+33
disjunction	$S \vee T$	+28
composition	$S \fatsemi T$	−1
existential quantification	$\exists S \bullet \ldots$	−7
negation	$\neg S$	−8
universal quantification	$\forall S \bullet \ldots$	−10
renaming	$S[y_1/x_1,\ldots,y_n/x_n]$	−11
hiding	$S \setminus (x_1,\ldots,x_n)$	−15
implication	$S \Rightarrow T$	−17
piping	$S \gg T$	−17
projection	$S \upharpoonright T$	−18
iff	$S \Leftrightarrow T$	−24

Figure 16. Popularity scores of the schema operators

Note that we explicitly asked people about schema renaming and piping even though they are not part of Spivey's notation [Spivey 1989]. A number of interviewees said that they did not use these particular aspects because they were not supported by the tool which they were using.

In addition we asked interviewees to indicate if there were any extensions to the notation which they used, this generated a list of fifteen items of which the use of a 'where' clause (four votes) was the most popular.

6.2 Comparison of Z with other formal notations

The participants in our survey had experience of many other formal notations, the ones which were mentioned were VDM [Jones 1980], [Jones 1986], CSP [Hoare 1985], CCS [Milner 1980], LOTOS [ISO/TC97/SC21 1987], RAISE [George & Milne 1991], HOL [Gordon 1985], OBJ [Goguen & Winkler 1988], [Gallimore *et al*, 1989], Larch [Guttag et al 1985b], [Guttag et al 1985a], and Abstract Machine Notation (AMN) [Abrial 1991]. In general, participants recognised that each formal notation is designed for a different purpose and that the appropriate one should be chosen for the specification in question. However, the support for structuring provided by Z was mentioned by many people as an advantage. Here we present a summary of the comments which people made when comparing Z with other notations. These remarks should be read for what they are — individuals' views.

By far the most oft mentioned notation which participants had tried was VDM. Of those who knew VDM there was a preference for Z due to the structuring mechanisms of the schema calculus. One person commented that 'VDM's explicit separation of preconditions, postconditions and invariants looks an advantage on the face of it, but the flexibility of Z is an advantage in the long run'. One project had started by using VDM, but after running into difficulties with structuring had switched to Z. Another project had also begun with VDM because one of the team knew it better, but they later changed to Z because they wanted to investigate it. The project left them with a positive view of Z. In general VDM was felt to be too much like a programming language whereas Z was more abstract and closer to mathematics.

CSP was recognised as being useful for concurrency. One participant commented that 'we choose Z or CSP depending on the view required'. Several people said that they would like to see support for concurrency brought into Z, and some had thought about combining CSP with Z.

Several participants had used CCS. It was described as more compact and easier to use than Z, but there was a recognition that Z was more abstract and this was seen as a good thing. One participant did say that CCS 'provides a model of how the system behaves which can be run in one's head'. Another liked the mutual recursion which CCS offers.

LOTOS was recognised as being valuable within its domain and one participant believed that LOTOS would have been useful for the specification in question. However, the lack of known proof work in LOTOS had been a limiting factor. Another participant who had worked with LOTOS had found it hard to use and thought that the specifications tended to be messy.

RAISE was acknowledged as offering a single language from specification down to pseudo-code, but was viewed as being much more complicated than Z.

HOL was seen as being a lot simpler than Z with a lot less syntax. This makes it less convenient than Z for specification but more convenient for proof. On the other hand Z, was seen as being better for structuring and abstraction.

Some participants found it impossible to compare Z and OBJ, but others did comment on the differences. Z was seen as more expressive and intuitive than OBJ. Contrariwise, other participants thought OBJ easier to learn because it wasn't necessary to get to grips with the logical underpinnings which are needed for Z. Two interviewees remarked that the tools for OBJ were better than for other notations which was an advantage.

Larch was seen as quite like Z and easy to understand. One participant had found it useful in specifying neural nets as it offered more freedom in this area than Z did.

AMN was preferred to Z by one participant, as it was seen to give better support for consistency and refinement proofs. The tool support for AMN was thought to be better than that for Z. AMN also won over Z for permitting a complete development in the same notation.

Perhaps we should not be surprised that nearly all of those participating in the survey preferred Z to other notations. Many of the participants were volunteers from the 1990 Z User Meeting or members of organisations participating in the ZIP project. Although it would be unwise to rely too much on the views of a few individuals, it is interesting to note that the availability of better tools was a factor in people choosing other notations.

6.3 Comparison of Z with structured specification methods

There was a wide-spread acknowledgement that Z could usefully be combined with structured approaches, and a number of interviewees had done this. Z has been used in conjunction with SSADM [SSADM 1986], [SSADM 1990], Yourdon [Yourdon & Constantine 1978], [Gane & Sarson 1979], [Semmens & Allen 1991], RTSA [Mellor & Ward], and CORE [Mullery 1979]. One person suggested that a good approach might be to use a structured method to break down a problem and then use Z to specify each part. The use of diagrams in structured methods was felt to be helpful and to aid understanding — an important point for specifiers to note!

Structured methods were recognised for the advice they give on approaching problems. One interviewee reported on a project in which the best Z specifications were written by a team which had the least maths background. This was believed to be because they had done Yourdon; they knew how to start, how to break down the problem, and how to spot important aspects of the system.

Another participant who had tried Z and SSADM together commented that there is an analogy between the two specification processes. It was felt that there are definite hooks to SSADM from Z, for example, in completeness checking. However, the need for people in the specification team to be familiar with both notations was acknowledged as a drawback.

RTSA was used on one project to design most of the system and then this was integrated with Z for those parts which required it. The two approaches did not mesh particularly well: the parts of the specification could not be composed in the same way in Z as for the diagrams. As a result once a particular sub-system had been identified then the whole of it would be specified in Z rather than trying to follow the data flow diagram description.

Several people wanted a formal basis for structured methods although they appreciated the method in the structured specification approaches. As one responder put it 'Z doesn't have a method, just a language. The others have a method, but no language'.

7. Conclusions

The future for Z looks promising. Companies that have tried it are pleased with the outcome and want to continue using it. A limiting factor in the uptake of Z is often the mistaken belief that it is too difficult and that no-one uses it for real projects. Those users we surveyed were able to write Z specifications after limited training, although availability of a local expert is important.

There is considerable interest in the use of Z in security related fields. Nevertheless, we have discovered a wide variety of real uses of Z which are not security or defence related.

Users found that writing in Z brought success even if they did not prove anything about their specification, nor even write down desirable theorems or proof obligations. Implementation from the formal specification was found to be enjoyable, with some people commenting that the use of the Z specification had made the coding task easier since it was clear what had to be done.

There is beginning to be some interest in proof at the specification level, although lack of tool support here appears to be a more limiting factor than for specification. Several companies are looking at refinement techniques, but there is a need for one which is integrated with Z.

We hope that the experiences we have reported here will help those who need evidence that Z really can be used gainfully on commercial projects.

Bibliography

[Abrial 1991]
Jean-Raymond Abrial. A refinement case study (using the Abstract Machine Notation). In Joseph M. Morris and Roger C. Shaw, editors, *4th Refinement Workshop,* Workshops in Computing, pages 51-96. Springer Verlag, 1991.

[Barden *et al.* 1991]
Rosalind Barden, Susan Stepney, and David Cooper. Report of a survey into the use of Z. ZIP document ZIP/Logica/91/094, Logica Cambridge Ltd., December 1991.

[Barrett 1989]
Geoff Barrett. Formal methods applied to a floating-point number system. *IEEE Transactions on Software Engineering,* SE-15(5):611-621, May 1989.

[Brownbridge 1990]
David Brownbridge. Using Z to develop a CASE toolset. In [Nicholls 1990], pages 142-149.

[Dick *et al.* 1990]
A. J. J. Dick, P. J. Krause, and J. Cozens. Computer aided transformation of Z into Prolog. In [Nicholls 1990], pages 71-85.

[Gallimore *et al.* 1989]
R. M. Gallimore, D. Coleman, and V. Stavridou. UMIST OBJ: a Language for executable program specifications. *The Computer Journal,* 32(5):413-421, 1989.

[Gane and Sarson 1979]
C. Gane and T. Sarson. *Structured Systems Analysis: Tools and Techniques.* Prentice Hall, 1979.

[Garlan and Delisle 1990]
David Garlan and Norman Delisle. Formal specifications as reusable frameworks. In Dines Bjorner, C. A. R. Hoare, and H. Langmaack, editors, *VDM'90: VDM and Z - Formal Methods in Software Development, Kiel,* volume 428 of *Lecture Notes in Computer Science,* pages 150-163. Springer Verlag, 1990.

[George and Milne 1991]
C. W. George and Robert E. Milne. Specifying and refining concurrent systems - an example from the RAISE project. In C. Carroll Morgan and James C. P. Woodcock, editors, *Third BCS-FAGS Refinement Workshop,* Workshops in Computing, pages 155168. Springer Verlag, 1991.

[Goguen and Winkler 1988]
Joseph A. Goguen and Timothy Winkler. Introducing OBJ3. Technical Report SRI-CSL-88-9, SRI International, 1988.

[Gordon 1985]
Michael J. C. Gordon. HOL: A machine oriented formulation of higher order logic. Technical Report 68, University of Cambridge, 1985.

[Guttag *et al.* 1985a]
J. V. Guttag, J. J. Horning, and Jeannette M. Wing. The Larch family of specification languages. *IEEE Software,* 2(5):24-36, 1985.

[Guttag *et al.* 1985b]
J. V. Guttag, J. J. Horning, and Jeannette M. Wing. Larch in five easy pieces. Technical report, Digital Systems Research Center, July 1985.

[Hall 1990]
J. Anthony Hall. Seven myths of formal methods. *IEEE Software,* pages 21-28, September 1990.

[Hoare 1985]
C. A. R. Hoare. *Communicating Sequential Processes.* Prentice Hall, 1985.

[ISO/TC97/SC21 1987]
ISO/TC97/SC21. LOTOS, a formal description technique based on the temporal ordering of observational behaviour. Draft International Standard ISO/DIS/8807, International Standards Organization, 1987.

[Jones 1980]
Cliff B. Jones. *Software Development: a rigorous approach.* Prentice Hall, 1980.

[Jones 1986]
Cliff B. Jones. *Systematic Software Development using VDM.* Prentice Hall, 1986.

[May 1990]
David May. Use of formal methods by a silicon manufacturer. In C. A. R. Hoare, editor, *Developments in Concurrency and Communication*, University of Texas at Austin Year of Programming series, pages 107-129. Addison-Wesley, 1990.

[Mellor and Ward 1986]
Stephen J. Mellor and P. T. Ward. *Structured Development for Real Time Systems.* Yourdon Press, 1985, 1986.3 volumes.

[Milner 1980]
Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer Verlag, 1980.

[Mitchell et al. 1991]
Richard Mitchell, Martin Loomes, and John Howse. Organising specifications: a case study. Technical Report BPC 91/1, Brighton Polytechnic, Department of Computing, January 1991.

[Mullery 1979]
G. Mullery. CORE - a method for controlled requirement specification. In *Proceedings of the 4th International Conference on Software Engineering, Munich,* pages 126-135, 1979.

[Nash 1990]
Trevor C. Nash. Using Z to describe large systems. In [Nicholls 1990], pages 150-178.

[Nicholls 1990]
John E. Nicholls, editor. *Z User Workshop: Proceedings of the Fourth Annual Z User Meeting, Oxford*, Workshops in Computing. Springer Verlag, 1990.

[Nix and Collins 1988]
Christopher J. Nix and B. Peter Collins. The use of software engineering, including the Z notation, in the development of CICS. *Quality Assurance*, 14(3):103-110, September 1988.

[Parker 1991]
Colin E. Parker. Z tools catalogue. ZIP document ZIP/BAe/90/020, British Aerospace, Warton, May 1991.

[Phillips 1990]
Mark Phillips. CICS/ESA 3.1 experiences. In [Nicholls 1990], pages 179-185.

[Semmens and Allen 1991]
Lesley Semmens and Pat Allen. Using Yourdon and Z: An approach to formal specification. In John E. Nicholls, editor, *Proceedings of the Fifth Annual Z User Meeting, Oxford,* Workshops in Computing, pages 228-253. Springer Verlag, 1991.

[Shepherd and Wilson 1989]
David Shepherd and Greg Wilson. Making chips that work. *New Scientist,* 1664:61-64, May 1989.

[Spivey 1989]
J. Michael Spivey. *The Z Notation: a Reference Manual.* Prentice Hall, 1989.

[SSA 1986]
Manchester NCC. *SSADM Manual Version 3*, 1986.

[SSA 1990]
Manchester NCC. *SSADM Manual Version* 4, 1990.

[Stepney *et al.* 1991]
Susan Stepney, Rosalind Barden, and David Cooper. Comparative study of object orientation in Z. ZIP document ZIP/Logica/90/046(3), Logica Cambridge Ltd., December 1991.

[Stepney *et al.* 1992]
Susan Stepney, Rosalind Barden, and David Cooper. A survey of object orientation in Z. *IEE Software Engineering Journal,* March 1992.

[Yourdon and Constantine 1978]
Edward Yourdon and L. L. Constantine. *Structured Design: Fundamentals of a Discipline of Computer Program and Systems Design.* Yourdon Press, 2nd edition, 1978.

A. List of participating organisations

We are grateful for the time and effort which the survey participants have given and for the enthusiasm with which they participated. They came from the following organisations:

BNR (Europe) STC Technology Ltd, BP, Brighton Polytechnic, British Aerospace, DRA ED RSRE Malvern, GEC Avionics, GEC Marconi Research, GPT Videotex and Secure Systems, IBM United Kingdom Laboratories Ltd, ICL Secure Systems Ltd, Imperial College London, IST, KBSL, Logica Cambridge Ltd, Logica Defence and Civil Government Ltd, Logica Space and Communications Ltd, Oxford University Programming Research Group,

Praxis Systems Ltd, Racal Research Ltd, Roke Manor Research Ltd, Secure Information Systems Ltd, SERC RAL, Sheffield City Polytechnic, Siemens Plessey Defence Systems, Southampton University, and York University.

B. Acknowledgements

This work was carried out as part of the ZIP project which is an IED initiative, project number IED4/1/1639. The part funding by the DTI is gratefully acknowledged.

Phase one of the survey was greatly assisted by our partners in the ZIP consortium who found people for us to interview. Project members from the PRG attended most of the phase one interviews and wrote up the raw results of about a dozen.

Disclaimer