Journal of Industrial Microbiology, 12 (1993) 251–255

Published by The Macmillan Press Ltd

Developing and distributing user-friendly application software

Robert L. Buchanan

Microbial Food Safety Research Unit, USDA ARS Eastern Regional Reseach Center, 600 East Mermaid Lane, Philadelphia, PA, USA 19118

(Received 23 February 1993)

Key words: Technology transfer; Predictive microbiology; User-friendly

SUMMARY

The adoption of new techniques in predictive microbiology by the food industry will ultimately be dependent on the development of user-friendly application software that makes it easy for non-research personnel to employ the mathematical models. Such applications should be an integral part of projects in predictive microbiology. Recommendations related to the architecture, speed, protection, testing, and distribution of application software are presented based on our experience in developing and distributing the 'Microbial Food Safety Pathogen Modeling Program.'

INTRODUCTION

Modeling has been an integral part of food microbiology research for decades; however, the transfer of these techniques to non-researchers has been very limited. This partly reflects the limited scope of the models available, and a lack of recognition of predictive microbiology as a distinct subdiscipline within food microbiology. However, a key factor has been a failure to provide the models in a form that can be employed easily. The objectives of this manuscript are to recommend that development of such applications be an integral part of modeling research, and to suggest some of the general characteristics that such applications should possess. These recommendations are derived in large part from our experiences in designing and distributing the 'Microbial Food Safety Pathogen Modeling Program' (PMP).

The PMP is application software that was developed as a means of introducing the food industry to the potential uses of predictive food microbiology, and acquiring feedback on the relative effectiveness of the models developed by our research group. Currently in its fourth version (release 3.1), the software automates the use of response surface models that were developed by the Microbial Food Safety Research Unit at the USDA ARS Eastern Research Center. This includes models for *Listeria monocytogenes* [2], *Shigella flexneri* [8], *Aeromonas hydrophila* [6,7], *Staphylococcus aureus* [3] and *Escherichia coli* O157:H7 [3]. Additionally, the software includes a model for *Salmonella* spp. developed at the Bristol Laboratory of the AFRC Food Research Institute [5]. The variables considered by each of the models are summarized in Table 1. Some of the characteristics and assumptions underlying the software, particularly the earlier versions, have been detailed previously [1].

Correspondence to: R.L. Buchanan, Microbial Food Safety Research Unit, USDA ARS Eastern Regional Research Center, 600 East Mermaid Lane, Philadelphia, PA, USA 19118

Reference to a brand or firm name does not constitute an endorsement by the US Department of Agriculture over others of a similar nature not mentioned.

DESIGNING SOFTWARE

Model validation

A wide range of microbiological models can be automated for easy use; however, the models selected should be properly validated. The first phase of validation is to use the model to predict new data combinations, generate the corresponding experimental data, and evaluate the

TABLE 1

Microorganisms and variables included in version 3.1 of the MFS Pathogen Modeling Program

Microorganism	Temp (°C)	Initial pH	NaCl (%)	$NaNO_2$ ($\mu g\ ml^{-1}$)	Atm[a]
Listeria monocytogenes	5–37	4.5–7.5	0.5–4.5	0–200	A+AN
Shigella flexneri	12–37	5.5–7.5	0.5–5.0	0–200	A+AN
Staphylococcus aureus	12–45	4.5–8.4	0.5–16.5	0–200	A+AN
Aeromonas hydrophila	5–42	5.3–7.3	0.5–3.5	0–200	A+AN
Escherichia coli O157:H7	5–42	4.5–8.5	0.5–5.0	NI[b]	A+AN
Salmonella	10–30	5.0–7.0	0.5–5.0	NI[b]	A

[a]Atmosphere: aerobic (A) or nitrogen-flushed/sealed (AN); both agitated.

[b]Not included as a variable in model.

performance of the model. Models are often developed in an iterative manner, where several cycles of data acquisition and model evaluation are used to generate models and subsequently increase their reliability.

If microbiologial media or other non-food experimental systems were employed to develop a model, the second phase of validation is to determine how well the model predicts the microorganism's behavior in various food systems. This can be achieved by direct acquisition of experimental data or through the utilization of quantitative data available in the scientific literature. In the latter case, care is needed to ensure that the literature values are pertinent and realistic, particularly when values for additional parameters that could affect the microorganism are not provided. As will be discussed further, it is often advisable to provide disclaimers both in the application software and the supporting documentation that describe how a model was developed and the limits associated with its use and interpretation.

User needs

Once a group of validated microbiological models are available, a critical step that should proceed the initiation of the development of a software package is defining who will be using the program and what are their needs (i.e., who are your 'clients'). For example, a single variable model may be adequate for predicting the growth of bacteria in fluid milk. This product is blended to be highly homogeneous, so its shelf-life is largely dependent on storage temperature. Alternatively, developing a software package that provides estimates of microbial growth kinetics for a range of products that are dependent on multiple variables will require more complex models and application programs. There are trade-offs associated with the different classes of models. While models that encompass multiple variables have greater potential applicability, they are more difficult to develop and tend to be less precise than those for a single variable.

Knowledge of the user's background is also needed to decide to what degree they should be allowed to alter the program or modify the initial model. However, in most instances it can be assumed that manual calculations and manipulations should be minimized, if not outright avoided. Typically, the more a program can be designed so that actions by the user are limited to inputting values or selecting predetermined options, the more user-friendly it will be.

Program format

The selection of programming format for developing application software is dependent on a variety of factors including the features desired, the size of the application software, the familiarity of the developer with different programming languages and options, the hardware available to the user, and the probable familiarity of the user with the programming approach.

In most instances it is no longer necessary to become expert in a programming language such as BASIC, FORTRAN, or PASCAL to write effective applications. A number of commercial spreadsheet or database programs have reached the point of sophistication that they provide very effective formats for developing predictive microbiology applications. Further, they often include features such as graphics capabilities that would require extensive knowledge of a programming language if a developer were starting from scratch. In the initial development of the PMP, the commercial spreadsheet program, Lotus 1-2-3 (Lotus Development Corp., Cambridge, MA, USA) was selected as the application format for several reasons including:

(a) A high degree of familiarity with the macros and command language associated with the software,
(b) the program had extensive graphics capabilities that could be readily incorporated into applications, and
(c) Lotus 1-2-3 is widely used so it was likely that PMP users would have access to that or a compatible spreadsheet program.

This latter factor limits the potential use of commercially available programs to write predictive microbiology applications. However, the increased availability of compiler programs for a number of the commonly used software packages eliminates much of this problem. For example, the latest version of the PMP was written using Lotus 1-2-3 and then compiled using Baler (Baler Software Corp., Rolling Meadows, IL, USA). This allows PMP 3.1 to operate as an independent program. This approach has the added benefits of compressing the size of the program and insuring that the equations and macros cannot be altered.

Speed

Even though a computer may be solving equations that would take a human weeks to calculate manually, people dislike waiting for a computer. Accordingly, a major design criteria should be the speed at which the program carries out its various subroutines. The working rule of thumb is that if a macro takes more than 30 s to complete, an effort should be made to find a faster alternative. There are several approaches that can significantly enhance the speed of applications:

(a) Operate the program from a hard disk instead of floppies. Typically, hard disks are > 10-fold faster. Writing 'install' batch files that automate the transfer of the application software to a hard disk facilitates this for users.
(b) Use a multiple file design. Keeping the size of any single file as small as possible greatly speeds calculations and related subroutines. For example, the latest version of the PMP is a set of 26 integrated files.
(c) Take advantage of macros and related design options that increase speed. For example, the command language associated with Lotus 1-2-3 allows the sequential display of screens and menus associated with the operation of various macros to be 'shut-off,' greatly decreasing the time required to complete the macro. Likewise, delaying or performing only limited recalculations are other options that can greatly enhance application speeds.

Protecting the program

When users think about program protection, it is usually in relation to safeguards incorporated into commercial software to prevent its unauthorized duplication. However, a more important consideration for application developers, particularly for those employing commercially available spreadsheet or database systems, is the internal safeguards that can be employed to prevent the unintentional alteration of an application's models or macros. Considering that the heart of any application is one or two mathematical models, even a small modification to these equations will drastically affect performance. Likewise, modifications to the underlying macros will disable an application.

Typically, there are several options available for enhancing protection when using commercially available software to design applications. For example, when developing the PMP, one of the first design decisions was to place all the macros in an isolated portion of the spreadsheet that would be unlikely to be altered if the normal macro sequence was disrupted. In addition, the macros were all formatted as being hidden such that they would not be displayed if that portion of the spreadsheet was accessed. Security was also enhanced by taking advantage of the protection option available with Lotus 1-2-3. Only those cells where the user must input data were unprotected. In the latest version of the PMP, the use of a compiler further elevates the level of security so that it is virtually impossible to inadvertently alter the application files. However, knowing how ingenious people are at getting around security systems, an integral part of our application package is our phone number so they can call for help.

Ensuring agreement with biological response

When designing subroutines that extend the flexibility of an application, care must be taken not to introduce systematic errors that do not reflect actual biological responses. A case in point with the PMP was the effect of initial inoculum size on predicted growth kinetics values. Biologically, inoculum size has little effect on the subsequent growth kinetics of the vegetative cultures [2–4,6,8]. However, as can be observed in the example depicted in Table 2, a straight substitution into the equations for calculating generation time (or lag phase duration) resulted in a systematic error. In this instance, the problem was overcome by assuming a fixed value for C. The details of assumptions underlying this modification have been described previously [1].

Ensuring proper use of models

Effective models that have been adapted for 'client' use have great applicability as a rapid means of estimating the behavior of microorganisms in food systems. However, there is a need to ensure that the models are used correctly. One of the key factors that must be re-enforced is that models provide 'first estimates.' These permit scientists to work more efficiently by assisting them in focusing their efforts. At least at this stage, the models are not a substitute for laboratory analyses, but instead are a means of allowing the laboratory to work more cost effectively by identifying priority areas. For example, being able to estimate the combinations of formulation variables that would be likely to prevent the growth of critical foodborne pathogens would greatly decrease the number of prototype products that would have to be tested when developing a new food product. If predictive microbiology is to gain credibility, it is critical that enthusiastic application developers do not overstate the effectiveness of their products.

TABLE 2

Comparison of the effect of using a floating versus a fixed value for the Gompertz C parameter on the subsequent calculation of generation time for *Escherichia coli* O157:H7 [3]
(Culture conditions: incubation temperature = 28 °C, initial pH = 7.5, NaCl content = 0.5%, $NaNO_2$ = 0 μg ml^{-1}, aerobic incubation)

Inoculum level (CFU ml^{-1})	Generation times (h)		
	Observed[a]	Floating C-value[b]	Fixed C-value[c]
80	0.52	0.44	0.58
795	0.67	0.51	0.58
10016	0.38	0.61	0.58
25165	0.66	0.66	0.58
100208	0.71	0.74	0.58

[a]Average of three replicates.
[b]Based on C = MPD − A. MPD = Log_{10} (Maximum Population Density).
[c]Based on grand mean of all C-values.

One of the ways that we have approached this problem is through the inclusion of disclaimers. Specifically, the first screen for initiating each of the models describes the conditions under which the model was developed, the citation if the model has been published, and a statement indicating that model performance can only be 'guaranteed' under the identical conditions under which the original data were generated. It is important that potential users be warned that the model is to be used for acquiring estimates, not final answers. It is worth noting that models developed using microbiological media tend to be inherently conservative. The use of effective disclaimers may also be important in relation to limiting the developer's liability.

Another factor that must be controlled to ensure proper use of predictive models is the selection of the variable values inputted by the user. The majority of predictive models available currently are only valid within the limits of the original experimental data. The models effectively interpolate, but using them to extrapolate should be vigorously discouraged. Particularly with the more complex response surface models, using values even slightly outside the experimental ranges may lead to highly erroneous results. We have found that the most straightforward way of ensuring that appropriate values are inputted is to state the permitted range of each variable next to its data entry box, with a clear warning that the values outside those ranges are not valid. An even more effective approach is to develop

subroutines that prevent the program from accepting values outside the specified ranges.

Finally, a key factor in effective use of the program is the instructions supplied with the package. All instructions should be clear, with care being taken to avoid jargon. The effective use of figures, tables and other visual aids can greatly enhance the usefulness of support documentation.

Consumer testing

After initial design and development of an application, we have found it very advantageous to 'consumer test' the software. No matter how effectively one tries to edit a program, predict problems or ensure that instructions are clear, there are always items that are overlooked. Further, there is a tendency for the developer to overestimate the familiarity of the user with the program. Such problems surface quickly when the application is consumer tested.

An important consideration is the selection of a test population that reflects the ultimate users of the software. Generally, the best test population is one that encompasses a broad range of experiences both in relation to the specific system that serves as the basis of the application and computers in general. Tasks and terms that are obvious to the developer may be foreign to someone who is a neophyte to computer operations. Acquisition of useful feedback from the test population can be enhanced by supplying simple questionnaires that help focus the reviewer in relation to specific attributes that you need to have considered. Of particular interest is the identification of any operation that results in the program 'hanging up.'

DISTRIBUTION OF APPLICATION SOFTWARE

Once completed, the next phase is the distribution of the software. An important decision that should be considered prior to release is the types of 'intellectual property rights' that are available to the developer. Unless a new algorithm was developed, the protection is generally restricted to copyrights. Specific advice on copyright requirements and limitations should be acquired from individuals with the appropriate expertise.

An accompanying decision is what type of protection, if any, the developer wants to use to prevent unauthorized reproduction. Personal experience indicates that most types of security can be overcome readily. Controlling reproduction may be better handled through licensing arrangements. Again, consultation with professional software developers is recommended. In the case of the PMP which is distributed free of charge, we simply ask that the program not be given to third parties. Instead, we request that current users supply other interested individuals with our telephone number so they can call to acquire their own copy.

One of the primary reasons for our request not to copy the program is that we maintain an active mailing list of people and organizations that have received the software. This is recommended for several reasons. First, this allows for automatic forwarding of new versions of the application as they become available. This has proven to be a great deal simpler than attempting to solicit new requests with each new release. Maintaining a file of recipients also allows corrections to be forwarded when a systematic or programming error is discovered after a version has been released. Finally, our active mailing list demonstrates concretely that there is interest in our predictive microbiology research. This can be highly beneficial when attempting to justify new research. The downside of maintaining technology transfer records is that it is time consuming. The mailing list for the PMP currently exceeds 500 entries, and it requires a substantial amount of clerical assistance to release a new version.

When preparing for a general release of a new application program, the scientists that developed the software have to be willing to 'sell' their achievement, a role in which they are not always comfortable. However, the majority of potential users of predictive microbiology applications may not be the individuals who regularly read a more traditional scientific journal. We have been aided significantly by our information services group. They helped us identify potential media to target, and then assisted us in the preparation of news releases and related announcements. With interest in computer applications expanding, several of the professional journals (e.g., *ASM News*) now review software along with the more traditional book reviews. We are also seeing the introduction of specialty journals specifically targetted to computer modeling. For example, *Byte* is a recently introduced journal that focuses on computer applications for microbiologists and microscopists.

CONCLUDING REMARKS

The importance of developing user-friendly applications that transfer to the food industry new technologies arising from predictive microbiology research cannot be overstated. Without such an effort, the results of this exciting work will languish as an interesting, but rather esoteric area of research. For this reason, our laboratory has made it a point to include the development of user applications as an integral part of any project in predictive microbiology. From a personal standpoint, the development of such applications has proven to be both challenging and rewarding. Considering the tremendous strides in our basic knowledge of mathematical modeling, coupled with the increased ease with which highly sophisticated applications can be developed through expert system shells, it seems virtually certain that during the next five years predictive microbiology and related computer applications will dramatically alter the way food microbiologists analyze, interpret, and present their data.

REFERENCES

1 Buchanan, R.L. 1991. Using spreadsheet software for predictive microbiology applications. J. Food Safety 11: 123–134.

2 Buchanan, R.L. and J.G. Phillips. 1990. Response surface model for predicting the effects of temperature, pH, sodium chloride content, sodium nitride concentration, and atmosphere on the growth of *Listeria monocytogenes*. J. Food Protect. 53: 370–376.

3 Buchanan, R.L., L.K. Bagi, R.V. Goins and J.G. Phillips. 1993a. Response surface models for the growth kinetics of *Escherichia coli* O157:H7. Food Microbiol. 10: (In press).

4 Buchanan, R.L., J.L. Smith, C. McColgan, B.S. Marmer, M. Golden and B. Dell. 1993b. Response surface models for the effects of temperature, pH, sodium chloride, and sodium nitrite on the aerobic and anaerobic growth of *Staphylococcus aureus*. J. Food Safety 13: 159–175.

5 Gibson, A.M., N. Bratchell and T.A. Roberts. 1988. Predicting microbial growth: growth response of salmonellae in a laboratory medium as affected by pH, sodium chloride and storage temperature. Int. J. Food Microbiol. 6: 155–178.

6 Palumbo, S.A., A.C. Williams, R.L. Buchanan and P.G. Phillips. 1991. Model for the aerobic growth of *Aeromonas hydrophila* K144. J. Food Protect. 54: 429–435.

7 Palumbo, S.A., A.C. Williams, R.L. Buchanan and J.G. Phillips. 1992. Model for the anerobic growth of *Aeromonas hydrophila* K144. J. Food Protect. 55: 260–265.

8 Zaika, L.L., J.G. Phillips and R.L. Buchanan. 1992. Model for aerobic growth of *Shigella flexneri* under various conditions of temperature, pH, sodium chloride and sodium nitrite concentrations. J. Food Protect. 55: 509–513.