The POD System of Interactive Composition Programs

by Barry Truax
Dept. of Communication Studies, Simon Fraser University
Burnaby, B. C., Canada V5A 1S6

I. INTRODUCTION

The POD system consists of a number of programs for real-time synthesis and interactive composition established by the present author at the Institute of Sonology, Utrecht (1972-73), [1] and Simon Fraser University, Vancouver (1973-present). [2] Related versions of the programs exist at other centers in Canada, and Europe. All run on minicomputers of various types (PDP-15, HP 2116, PDP-11, Nova 3) and are accessible to users with little computer experience. The basic level of the programs utilizes real-time monophonic synthesis (either fixed wave-form synthesis with amplitude modulation, or the frequency modulation method developed by John Chowning at Stanford), [3] whereas a higher program level calculates non-real-time pressure functions which can be mixed and output at high sampling rates.

The compositional model embodied in the POD system can be represented in a block diagram such as that of Fig. 1. The user strategy, external to the program, works with the mental representation that the user has of a possible goal structure, and that of the present structure derived from hearing the sound synthesis result; this feedback is shown by the dashed line in Fig. 1. The interactive nature of the program lies in the user exercising control over various parts of the program to obtain successive modifications of the synthesized structure. That is, the user works on both the sonic and syntactic levels within the program, the principal task being to establish the relationship between the two (as shown in a protocol analysis below). The semantic level of operation is that of the user evaluating interim results and modifying the strategy for obtaining a satisfactory goal structure.

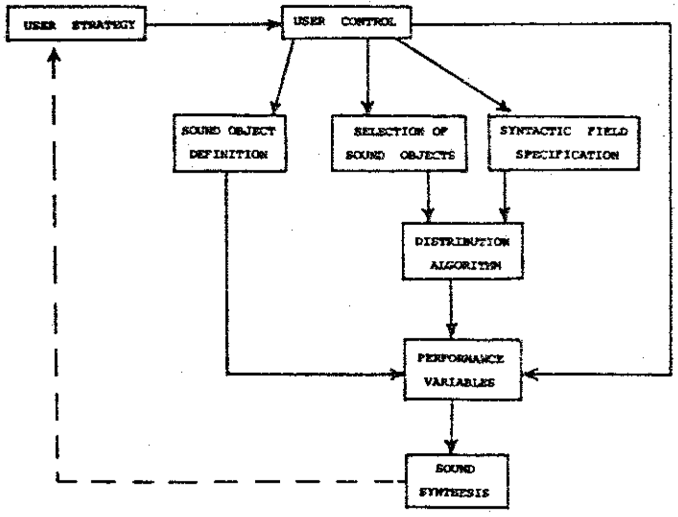

FIGURE 1. Compositional Model of the POD system.

II. COMPOSITIONAL STRUCTURE

The internal characteristics of the program are shown in Fig. 1 as four interconnected boxes identifying

1) Sound object selection;
2) Syntactic field specification;
3) Distribution algorithm;
4) Performance variables.

The data structure utilized by the program demonstrates a hierarchy of levels. The levels of generality incorporated within the program may be shown as:

Composition Section and Variants
Performance Variables
Distribution of Events
Sound Object
Event

This representation shows that the event is the basic perceptual unit posited by the program. In terms of formal data representation, the event is determined by four numbers representing frequency, time delay, maximum amplitude, sound object number. It should be emphasized that these values are not intended to dictate an unambiguous sounding structure. In fact, these parameters are kept as general as possible in order to maximize the number of possible interpretations through the use of performance variables. Time delay, as will be shown below, allows the greatest variety of interpretations to be implemented.

The sound object, as the next level of generality, is the data representation of the sounding object without those parameters determined in the event data, i.e. frequency and maximum amplitude. In other words, the sound object is a set of values applying to the variables of the synthesis program. In terms of frequency modulation synthesis, the sound object comprises the following parameters:

1) amplitude envelope (in terms of attack, steady state and decay times);
2) ratio of carrier to modulation frequencies (to determine the kind of spectrum, i.e. the set of available partials in the timbre of the sound);
3) the maximum modulation index (i.e. the strength of the spectrum; the index determines the largest

---

This article is excerpted, with permission, from a larger article entitled "A Communicational Approach to Computer Sound Programs," *Journal of Music Theory*, Vol. 20, No. 2, Fall, 1976.

number of partials significantly present in the spectrum, and their respective amplitudes);

4) the modulation index envelope (i.e. the temporal behavior of the spectrum as determined by the change of the modulation index within the sound).

It should be emphasized that the sound object could equally well be specified by an entirely different set of parameters in the case of a different synthesis program. That is, the compositional strategy is independent of the synthesis model being used, although the user's strategy presumably is not. Since the sound object data may be changed independently of the event specification, the sound object may also be regarded as a performance variable; however, since it applies only to a certain class of events and not to the entire structure, it remains at a lower level of generality.

The next level of data representation is that of the distribution of events. This level also includes the selection of the sound object to be applied to specific events. That is, this level involves syntactic relationships between one event and the next, as well as structural characteristics of the large-scale form. At this level, density is the most important perceptual and structural unit of the program. Density introduces the basis of time structure within the program, a time structure that includes the entire distribution of events; that is, the notion of density, and changes in density, is only relevant when considering the entire sequence of sounds as a complete structure. Thus, density links the perceptual immediacy of the event with the structural totality.

Density is the major variable of the Poisson distribution. This distribution, mathematically expressed by the Poisson equation, describes any random arrangement of so-called independent events, that is, where one event is not causally related to another. The Poisson distribution applies in cases where the density of events is sufficiently low (less than 10 to 20 events/second), such that separate events may be identified, since a large density of events merges into a continuous distribution, just as high sound density results in a fusion of events into a continuous texture.

The syntactic field is constructed primarily as a frequency/time field, and secondarily as an amplitude/time field. By handling the problem in this way, it is assumed that frequency and maximum loudness are parameters of sound sequences that may be perceived in a statistical manner, that is, by the size and distribution of their range over longer sequences, not only by the relation between one event and the next. The user considers the structure of the entire distribution during its composition from beginning to end, and thus deals with its overall form and structure. This field structure is somewhat theoretical, however, in that specific details of sound and performance are not being considered; instead, the result of the field calculation is a set of point events that await further information (performance variables) as to how they should be realized in sound and time.

The variables specified by the user for the syntactic field and the object selection are:

1) Density: specification ranging from simple linear change to complex density variations.
2) Frequency mask: areas of the frequency/time field are blocked out (as in Fig. 5) such that events may only occur within the boundaries of the mask. The duration of the mask is the theoretical duration of the sequence being composed.
3) Amplitude selection: the choice of methods for controlling the loudness of each event range from aleatoric to tendency mask (see Fig. 7) and sequences; default case: maximum values throughout.
4) Sound object selection: rules for assigning a sound object to a given event range from simple aleatoric, through weighted aleatoric (ratio), time-varying weighted aleatoric (with a tendency mask, see Fig. 5) to sequential and permuted sequential choice.

All of these levels of specification consist of rules applying to the entire distribution of events. They determine the overall form and structure, and at the same time (with sound object selection) determine the relation between the sonic and syntactic level, that is, how the sound repertoire is assigned to the events in the distribution.

The specific data for each event is determined by the distribution algorithm, of which the Poisson theorem calculation is the most important part. The use of this theorem is both appropriate and useful on the following grounds:

1) Generality: as described above, the theorem applies to a random distribution of independent events. When the theorem is used to generate events, its use implies that relationships between events will only be patterned by the listener and/or determined by other means expressly used by the composer. In this sense, it is neutral: no *a priori* relationships between events are involved.
2) Structural unity: since the burden of specifying the details of individual events is dealt with by the Poisson algorithm, the user is free to concentrate on structural design. In practical terms, a minimal amount of data is required to generate a potentially large number of events, and thus the time until feedback of results occurs is greatly reduced. Furthermore, the user does not have to impose more control on the structure than desired at any moment; in fact, control is imposed only where the user wishes and to the degree desired, other details being left to the programmed algorithm.

3) Variants: since random numbers are involved in the calculation, random variants of a given structure may be generated by simply changing the start number of the random generator and recalculating the structure, the details of which will be different while retaining the same overall structural characteristics. The user may generate structurally equivalent distributions, and may choose those that are the most satisfactory, or else various versions may be created with additional compositional uses in mind.
4) Field unity: the Poisson distribution allows events to be determined in a two-dimensional field, as well as that of a single dimension (time). Therefore, it is not necessary to control pitch independently of time, although most other compositional methods take this independence for granted. However, one of the advantages of the POD method is that a single process determines both coordinates, ensuring a coherence of pitch and time dimensions.

A still higher level of generality used in the POD programs is that of the performance variable. A performance variable is simply a rule for the interpretation in time and sound of a distribution of events. A set of such variables allows a distribution to be performed in a variety of ways, and often the use of performance variables amounts to an optimization procedure for the user. Since performance variables may be implemented without recalculation of the distribution, their effects may be quickly evaluated. Both the learning potential and the practical efficiency of the system are greatly increased by the incorporation of these variables into the program structure.

Performance variables achieve their high level of generality because they are systematic interpretations of a general syntactic structure (the distribution) which has not been made specific in time or sonic interpretation. They achieve their high level of pragmatic value because they allow (together with real-time synthesis) an effective optimization (or "hill-climbing") procedure for the user, as opposed to

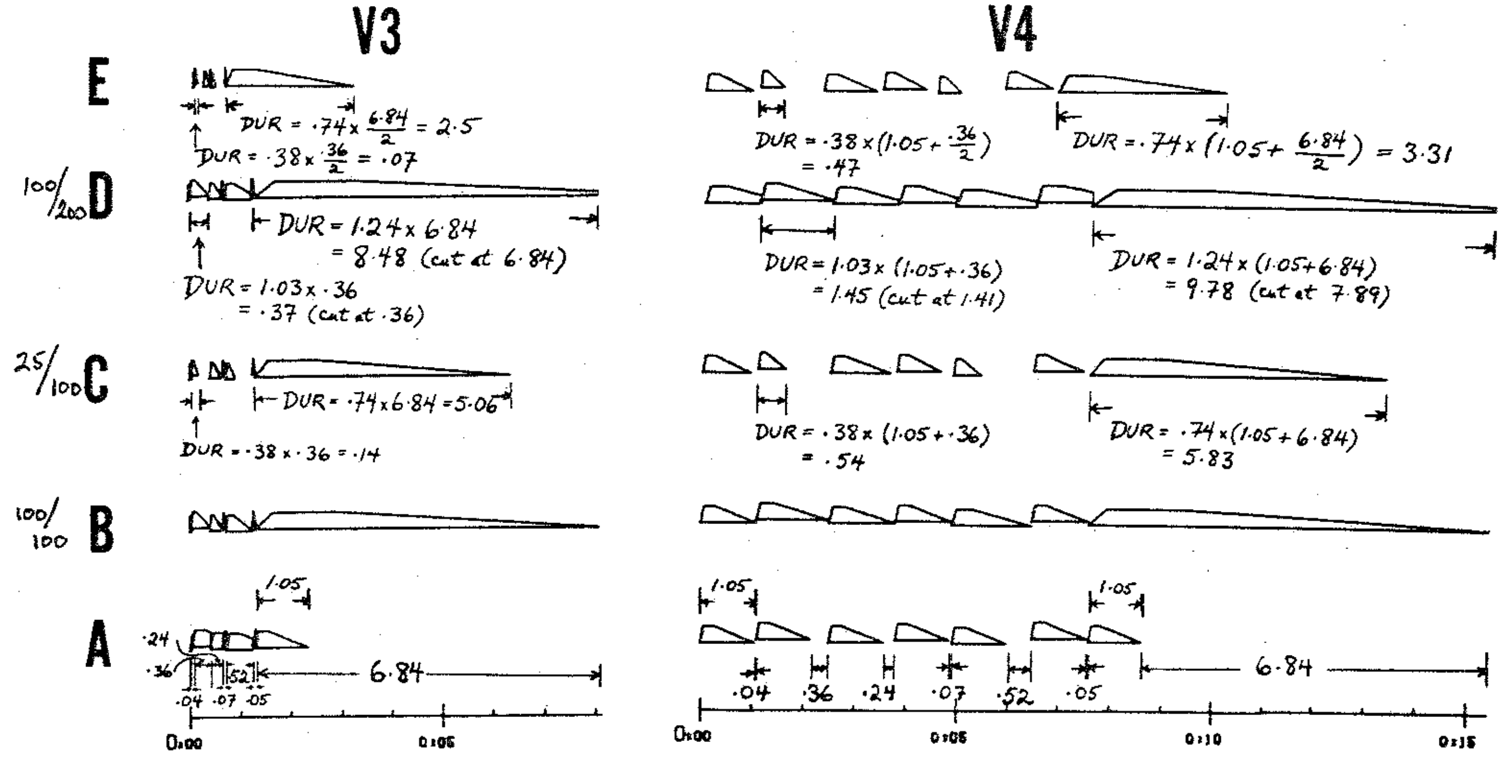

FIGURE 2. Envelope and Time Delay Modes. Seven events calculated by the Poisson algorithm have the series of times values: 4, 36, 24, 7, 52, 5, 684 (all in 100ths of sec.). The diagrams show a variety of interpretations of these basic time values with a single envelope (with attack, steady state, and envelope values of 5, 20, 80 respectively). The two basic time modes V3 and V4 treat the Poisson time values as entry delays (at left) and time delays (rests) at right. The five cases A-E show the following possibilities: A) normal sequence for V3 and V4; B), C), D) the Q5 scaling limits are shown at the left of each sequence. Note that for the 100/100 case, the envelopes are scaled to fit exactly into the available time. In C) the values of 38% and 74% are chosen for the 2nd and 7th events respectively within the 25/100 limits. In D) the values are 103% and 124% within the 100/200 limits. All envelopes in D) exceed or equal the available time, at which point they are cut off to make way for the next event. E) shows the effect of the speed factor (V1) combined with the scaling (Q5) of 25/100. With the speed doubled (V1 = 50), the entry delay with V3 is halved, and with V4 the rest is shortened; the scaling takes place on the basis of the new total time available.

the test-and-discard approach found in most systems.

Besides the sound object data which has already been described, the performance variables used in the POD system are:

1) Speed of performance, based on the norm of 100. This speed factor applies to the Poisson-calculated time delays, and thus its effect depends on the time mode used, as described below and shown in Fig. 2.
2) Direction of performance: forwards or with events in reverse order.
3) Time delay/envelope modes (V3,V4): the Poisson-calculated time delays may be interpreted in two complementary ways: i) V3 mode: as *entry delays*, that is, as the time between successive attack points; the envelope is cut off if not completed. ii) V4 mode: as *time delays*, that is, as the "rest" time between the end of one event and the beginning of the next; the envelope is uninterrupted. As can be seen in Fig. 2, the V3 mode realizes the Poisson time structure precisely, but may interrupt the events; the V4 mode extends the Poisson time structure by not interrupting the envelope whose duration (specified with the sound object) now contributes to the overall performance duration. Since the speed factor applies to either the entry delay (V3) or time delay (V4), its effect, in the former case, will be to adjust the tempo of the performed structure in the conventional manner, whereas in the latter case (V4), it will control the influence of the random (i.e. Poisson-calculated) rests. If the speed factor is zero in V4 mode, the performed structure depends entirely on envelope durations which, if proportionally specified, will produce a highly deterministic, even metric, rhythmic structure. Other speed factors will bring a controllable degree of randomness into this rhythmic pattern.
4) Envelope scaling: envelopes may be scaled to a certain percentage of the entry delay time available for each event. The range of percentages, within which a random choice is made, is specified as two numbers between 1 and 200%. The effect of various ranges can be seen in Fig. 2. Note that the entry delay with V3 is the Poisson-calculated time delay, whereas with V4 it is the combined envelope duration and Poisson delay. Envelope scaling effects an articulation of the event comparable to conventional *staccato* or *legato* markings.
5) Time delay limits: a range of acceptable time delays may be specified; those falling outside the range are converted to the limit value. The limit is compared with the Poisson-calculated time delay after it is scaled by the speed factor. With time delays less than the lower limit in the V3 mode, the event is omitted and the time added to the previous event.
6) Synthesis-related variables: in the case of frequency modulation, such a variable is the choice of treating the Poisson-calculated frequency as either the carrier or modulating frequency.

The compositional section or variant is the final stage of activity within the basic POD program, and is also the highest level of generality within the system. The compositional section comprises the data for a distribution of events, a repertoire of sound objects, and the performance variables required to realize the structure. This data may be stored in a disk file by the user, added to an existing file, extended with more sections, or transferred to magnetic tape immediately. It may also be retrieved at a later time for extension, modification, or performance. A structural variant may be simply the random variant described above, or else a structure calculated with different syntactic specifications. The performance variant is simply the same theoretical structure realized with a different set of performance variables. Both kinds of variants are useful for polyphonic mixing or multichannel recording, and their systematic generation may become part of a larger-scale compositional strategy.

Finally, any distribution created with POD6 may serve as input to related programs that process the data further. One such program translates the frequency and time values into a coded form of conventional notation that facilitates transcription (see Fig. 3 for a score example). Further acoustic manipulations are carried out by POD7, a non-real-time synthesis program that calculates and stores pressure function data on digital magnetic tape, and subsequently retrieves it for synthesis at a fixed sampling rate. This facility allows envelope overlap, a limited mixing facility, binaural localizaiton in two-channel output and digitally-calculated reverberation.

As a theoretical model, the POD system resembles a potential MUSIC V type system with a specific set of compositional subroutines and specialized synthesis instrument. As a real facility, the difference lies in the communicational model that characterizes it. That is, the system taken as program structure *and* user environment is vastly different from even an enriched MUSIC V system, principally in being an interactive system operating with real-time synthesis capability. Moreover, since it runs on a minicomputer system, accessibility and running cost are considerably more favorable.

The POD system incorporates strong, specialized strategies for both synthesis and composition, hence its limitations and its potential. These strategies are in recognition of the fact that most composer-users do not possess the detailed numerical knowledge required to use a generalized system effectively, and yet they are capable of quickly judging the wellformedness of acoustic results, and learning control strategies to modify results when the feedback time is sufficiently short. The strong, specialized program, then, allows the composer to work within a system already structured at every level in a manner conducive to his activity, instead of requiring such a structure to be imposed by every user on a basic, generalized facility.

The danger of a specialized system becoming usable by only a single composer needs to be avoided. For such a system to remain strong it must preserve a structure general enough to satisfy the needs of many different users with different types of problems. The POD system (having been used in more than a dozen compositions in the last 3 years by various composers) strives toward such a generality by implementing levels of control applying to structural characteristics of statistical distributions of sound. It implies that a wide range of contemporary musical techniques have something in common at the level of control structure, even when the content or message level is diverse. To develop such a model that can be shared by composers is in direct opposition to the notion of the composer's activity as irrational, non-generalizable, non-teachable, and non-programmable. It points to a competence shared among music users, but recognizes the diversity of strategies devised for achieving different performance goals.

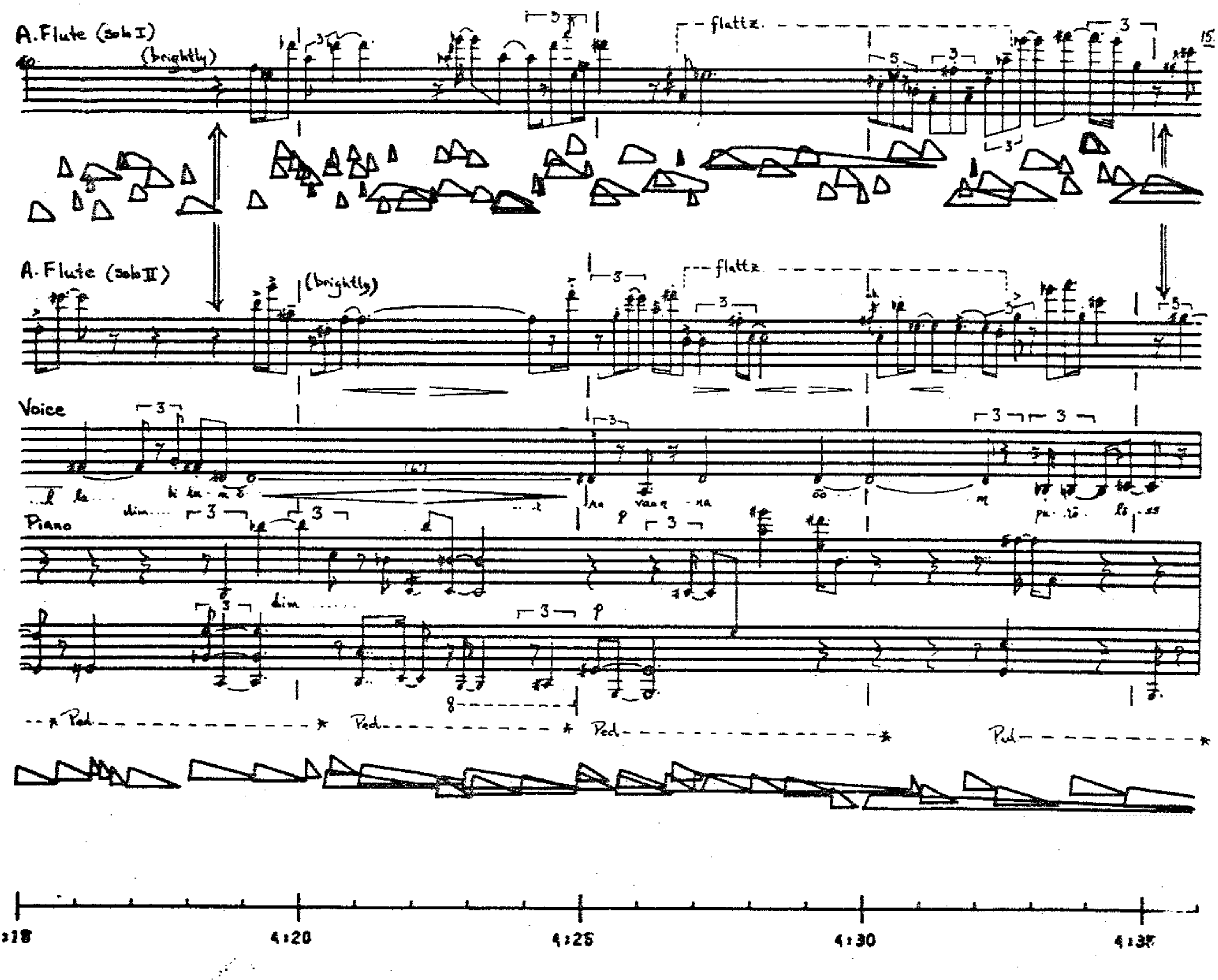

FIGURE 3. Excerpt from *Trigon* (1975), for alto flute, voice, piano, and computer synthesized tape, by Barry Truax. The two flute parts are random variants of the same structure. The graphic envelopes represent tape sounds.

III. POD PROTOCOL ANALYSIS

An informal protocol analysis of a compositional session with a POD program is presented here, first in order to clarify the structure of the system as described above by showing it in use, and secondly, to show the theoretical importance of protocol analysis as a data source for music theory.

Instead of listing the codes and data used by the user, each request is summarized in Table 1 by a short explanation of its effect. Only those commands having an effect on the process are listed, with those resulting from typing errors or extraneous factors omitted for simplicity. Also, the repeated sound tests are not made explicit as to the individual parameter changes requested by the user. The special situation created by this test-and-modify task environment will be discussed below.

The protocol steps enumerated in Table 1 may be grouped into the following operational sections:

A) Steps 1-12: Determination of the sonic repertoire beginning with spectral envelopes for which sets of suitable synthesis parameters are found.
B) Steps 13-16: Initial syntactic field specification and calculation. This includes the frequency/time field, object selection with a tendency mask, and density specification per mask segment.
C) Steps 17-26: Small modifications of the initial structure and its performance.
D) Steps 27-31: Respecification of final section of the syntactic field and the adjustment of beginning section.
E) Steps 32-38: Introduction of maximum amplitude/time field and adjustment of data.
F) Steps 39-49: Final optimization of structural and performance variables leading to a completed section being stored.
G) Steps 50-52: Creation and storage of a random variant.

The user's principal strategy follows a heuristic search model, first on the sonic level, then at the syntactic level, including the sonic-syntactic relationship, and finally at the level of the entire structure. Although work on all levels begins more or less independently, the optimization of the interrelationships between levels proceeds in parallel throughout the entire session. However, a closer examination of the process reveals that the sonic and syntactic levels are not conceived independently. In fact, the entire problem session seems to grow out of an idea gener-

TABLE 1. POD6 composition protocol (simplified), March 13, 1975, experiment T20-C, D/1; subject: B.T.

1. Specify 4 spectral envelopes (Fig. 4).
2. Repeated sound test with each spectral envelope.
3. Specify 5th spectral envelope (Fig. 4).
4. Repeated sound tests with 5th envelope.
5. Redefine 5th spectral envelope (Fig. 4).
6. Repeated sound tests with 5th envelope.
7. Specify 5 sound objects with 5th spectral envelope.
8. Repeated sound tests with 2nd spectral envelope.
9. Specify 5 sound objects with 2nd spectral envelope.
10. Repeated sound tests with spectral envelopes 3 and 4.
12. List sound objects.

13. Declare 20 objects available with selection by tendency mask (Fig. 5).
14. Specify Poisson field with mask (Fig. 5).
15. Specify density for each frequency mask section (Fig. 5).
16. Calculate distribution of events.

17. Adjust speed factor to 75%.
18. Perform (V3 mode assumed).
19. Set lower time delay limit to .08 sec.
20. Perform.
21. Get density analysis of distribution.
22. Lower the density for sections 1 and 2.
23. Recalculate and perform (twice).
24. Reduce object selection mask data for section 1 (allow 2 objects only).
25. Recalculate and perform.
26. Lower density for section 5.

27. Redefine Poisson mask (frequency/time) (Fig. 6).
28. Get density information and respecify (Fig. 6).
29. Recalculate and perform.
30. List object selection mask and specify new values (Fig. 6).
31. Recalculate and perform.

32. Specify and list amplitude tendency mask (Fig. 7).
33. Calculate amplitude values and perform.
34. Change first section of amplitude and frequency masks (increase range at end).
35. Increase density slightly for 2nd section and list.
36. Recalculate and perform (twice).
37. Change sections 4 and 5 of amplitude mask (make louder) (Fig. 7).
38. Recalculate and perform.

39. Reduce density slightly for section 2.
40. Perform (twice).
41. Change frequency mask, 1st section expanded slightly.
42. Recalculate and perform.
43. Scale envelopes to between 80 and 150% of entry delays.
44. Perform (twice).
45. Change random start number.
46. Recalculate and perform (twice).
47. Adjust amplitude tendency mask, 2nd section, to begin softer.
48. Recalculate and perform (twice).
49. Complete and save compositional section.

50. Change random start number.
51. Recalculate and perform (twice).
52. Complete and save compositional section.

ated by the spectral envelopes. They are specified first, then sound objects are developed that conform to their character, and finally a suitable syntactic field is created, modified, and optimized in which these spectra can meaningfully interact. The spectral envelopes and their related sound objects remain unchanged after the initial specification: instead, it is the syntactic field that undergoes a series of developments to accommodate the sonic repertoire.

Therefore, the principal problem dealt with in this protocol appears to be: the creation and optimization of sonic and syntactic structures appropriate to a set of initial spectral envelopes. The general method used to solve the problem is: generate an initial set of data, test, evaluate, and optimize. In this particular case, the mental activity that is the least observable in the protocol is that connected with the generation of the initial set of data. In each case, this data (Fig. 5) reveals in its complexity that a considerable amount of design work has already taken place. The spectral envelopes are complex, the initial object selection mask detailed, the initial frequency and amplitude masks are reasonably varied. Each of these represents the final state of the sub-process which has determined them, external to the program. It should be pointed out, however, that this process could equally well have taken place within the framework of the program in a series of steps proceeding from very simple data or very loose controls through increasingly complex data and increasingly specific controls. In fact, this is likely to be the procedure of the novice user, and it is apparent that if the program does not allow such progressive learning steps (i.e. concept formation processes), users will probably never reach the stage where an efficient use of the program is possible. If the purpose of the protocol generation were completely theoretical, constraints on the user's strategy could be imposed such that any data structure would have to be evolved from the simplest possible state to its final state such that all intermediate stages could be observed.

Having summarized the general nature of the problem-solving activity in this example, we now proceed to a more detailed analysis of selected aspects of the process:

1) **Initial constraints**: How much of the user's strategy was given at the beginning? That is, to what extent can it be inferred from the protocol that the present activity was constrained by a higher level plan? Any data that appears without a perception-based precedent may be interpreted as potentially arising from such an external constraint. In this protocol, examples of such data could be: i) the spectral envelopes (Fig. 4); ii) the proportions, range or durations of the tendency masks; iii) the speed factor (75%, since it was effected *before* a first result was heard). On closer examination, the spectral envelopes appear less certain as given data. The fifth one was modified and developed after the initial four and therefore is less likely to be a given quantity. It is also significant that the first envelope, an exponential curve) perhaps the most common type of spectral change, found in percussive sounds), is specified and never used except in single comparisons with other envelopes. Also suggestive of its normative role is the way in which its basic shape is modified in the second envelope to a sharp impulse form, then extended through the three-impulse form of the third, and the multiple impulse form of the fourth spectral envelope. Envelopes 2, 3 and 4 are designed as variations on the basic exponential shape, and envelope 5 as the time-reversed contrast to it. Further, in the final structure, envelopes 2, 3, and 4 are used together, as shown by the tendency mask

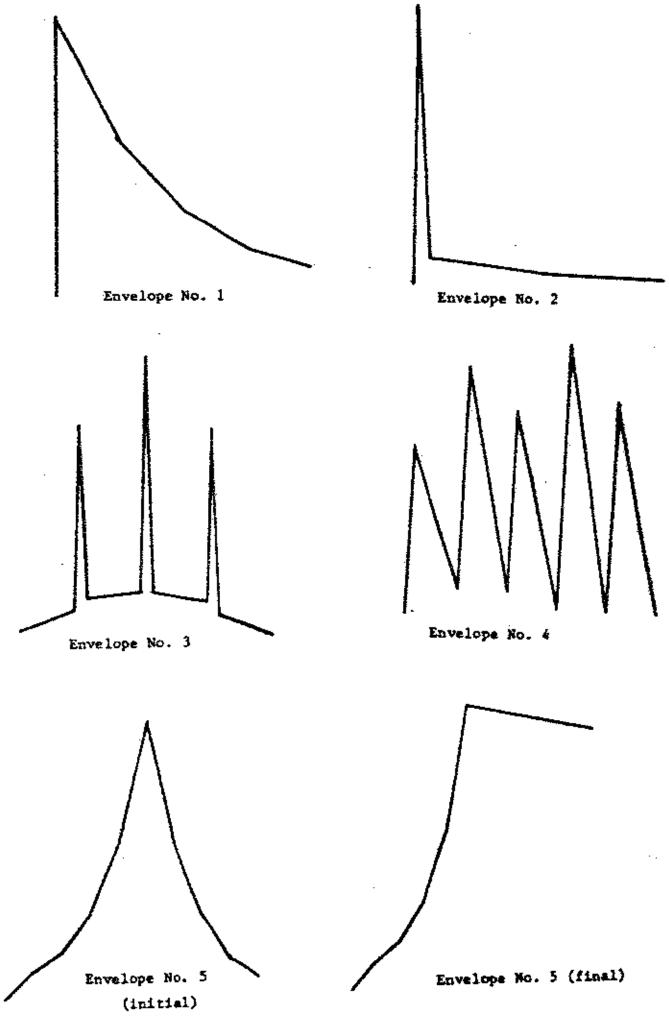

FIGURE 4. Spectral Envelope (of modulation index): Protocol Example.

(Fig. 6), whereas envelope 5 appears alone in the middle section. Whether given or developed, the set of envelopes clearly form a set of initial ideas complete with comparison/contrast implications. A comparison of other protocols in this series, not dealt with here, shows that the time proportions and speed factor of the present distribution are systematically related to all others in the series, and therefore may be considered as constrained by a higher level compositional strategy.

2) Sub-tasks: The efficient solution of any problem involves its being broken up into sub-tasks, each of which is independently soluble. The sub-task itself may comprise a set of simple sub-tasks, simple in the sense that each operation is trivial, but that together, they form a solution to the entire problem. The POD program assists this process of sub-task division in at least two ways. First, the overall structure of the program as outlined above divides the problem space into independent sub-tasks of sound object generation, syntactic field specification, performance variables, and so on. This division allows each sub-task to be carried out and modified spearately from the others. Secondly, the individual steps in each sub-task correspond to individual requests to the program in the form of options. Each option is mnemonically coded and performs a simple operation, such as data input, single or group data modification, data listing, distribution analysis, etc. The options are designed to handle the user's needs efficiently for the required tasks. Many have been implemented on the basis of user experience with the program with specific compositional problems.

The two main sub-tasks found in the current protocol are sound object generation and syntactic field specification. The latter is divided into further sub-tasks involving the frequency/time field, density specification, amplitude/time field and object selection. Some of these are specified in the form of masks, as shown in Figs. 5-7, where a choice of values along the vertical axis is allowed only in the circumscribed areas. This technique allows effective time-dependent choices to be easily specified.

The use of the tendency mask for sound object selection is the least obvious and needs further explanation. The vertical scale (as in Fig. 5) is divided into a number of equal units corresponding to the number of sound objects declared available. In Fig. 5, for instance, the percentage scale is divided into 20 parts for the 20 objects, with each 5% unit corresponding to a single object. Percentages are chosen from within the mask area, thereby indicating which object has been selected. For example, a percentage chosen between 1 and 5% selects the first object, between 6 and 10% the second object, and so on. This method allows controlled statistical choices varying in time to be made.

The use of the tendency mask in this case suggests that the user does not wish either random or completely deterministic choice to determine the distribution of the sonic repertoire within the syntactic field. Instead, the tendency mask (see Fig. 5) refers to each family of 5 sound objects (each associated with one spectral envelope) almost independently. In the first selection, choice is allowed from only the objects with envelope 2, then in the following sections, from those with envelope 5, then from those with envelope 3, then a transition from 2 to a mixture of 2 and 3, and finally a single object with envelope 4. The final configuration (Fig. 6) differs slightly in that only a single object is allowed at the beginning, and in the second half, there is a transition from objects with envelope 2 through those with 3, ending with those with 4. Since envelopes 3 and 4 have been generated together (protocol steps 10 and 11), this latter transition is hardly surprising.

However, the contrasting envelope (5) always remains isolated, and is never allowed to mix with the more percussive sounds. A special frequency range, as well as density and amplitude data, are also associated with this spectral envelope. These undergo practically no change from beginning to end, indicating that their initial design was satisfactory. On the other hand, the beginning section and the final one in particular, undergo numerous small changes of density and range before they are accepted. The choice of opening section, for instance, goes from the entire family of sound objects allowed in a narrow frequency range, to a single object distributed in an expanding range. In other words, the user's experimentation with the structure showed that a wide variety of sonic repertoire in a narrow frequency range was less effective as a beginning than a single sound object heard in a frequency range expanding from a semitone to a fourth. Similar changes were also effected in the final section regarding the variation of frequency range.

Performance variables are not varied to a great extent in this protocol. This is not because they are unimportant, but rather because the type of sound and syntactic field used was probably readily adaptable to the V3 time mode. Since envelopes 2, 3, and 4 are essentially percussive, these sounds with sharp attacks are not in danger of being marred if interrupted, except in the case of very short entry delays, which in step 19 are prevented by the lower limit being set to .08 sec. Late in the protocol, step 43, envelope scaling is introduced in the range between 80 and 150% of the available entry delay. This scaling prevents abrupt staccato articulation, and generally tends to smooth the attack/decay transition between events.

Note also that in step 45, the random start number is changed. Despite the fact that almost all changes have been made in the structure, the change in details

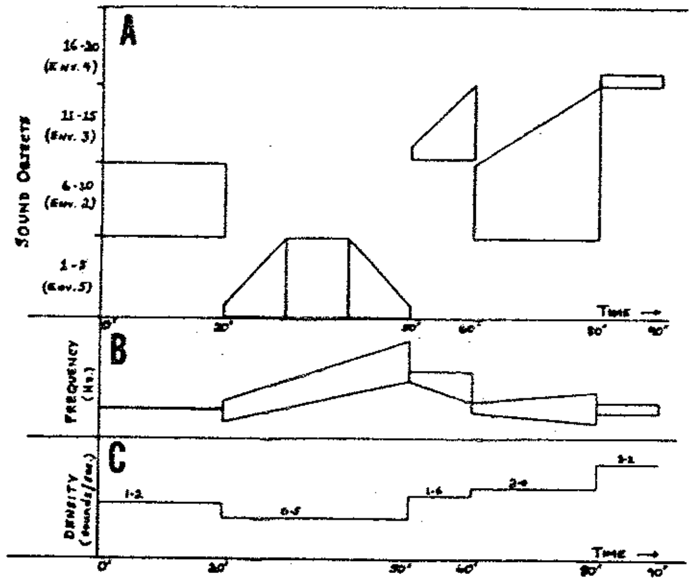

FIGURE 5. Protocol example: initial values of control structure for syntactic field. A) Object selection tendency mask for time-dependent choice of objects and related spectral envelopes at left; B) frequency mask; C) density variation.

at this stage appears to be acceptable. That is, the user is satisfied with the details of a different structure as long as the carefully worked out structural specification remains the same.

3) Sound object generation: the program allows the compositional structure to be by-passed such that the user can work with a single synthesized event. Particularly in the case of novel synthesis methods and parameters, this facility, with its rapid acoustic feedback, is very important for acquainting the user with the available sonic repertoire. After each sound is heard, the user may change a single parameter and hear the new result with all other parameters remaining as before.

This type of user activity is substantially different from the compositional kind in that many more decisions per time unit are made by the user and recorded in the protocol. Only in this case can it be said that short-term memory (STM) plays the predominant role in the user's mental activity, and therefore this type of protocol is extremely valuable as a record of decisions based on STM representations. [4]

It should be emphasized that the above constitutes only an informal and partial protocol analysis. As formulated by Laske, a complete analysis includes the following sub-problems:

1) setting up a task environment;
2) defining a basic problem space;
3) obtaining encoded data concerning human behavior in the form of protocols;
4) (from explicit records of behavior) determining the (augmented) problem space(s) actually used by subjects;

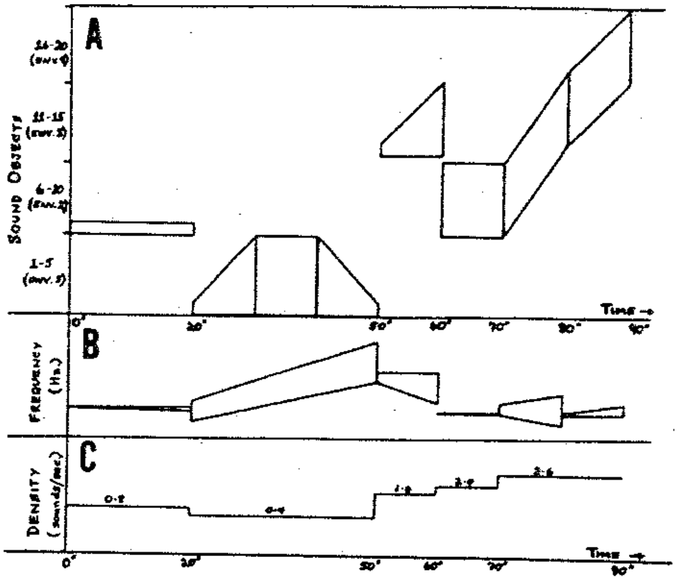

FIGURE 6. Protocol example: final values of control structure for syntactic field. A) Object selection tendency mask as in Fig. 5; B) frequency mask; C) density variation.

5) representing the search of subjects in their problem space(s) dynamically by way of a problem-behavior graph;
6) hypothesizing an ordered set of productions (program) which constitutes a model of the protocolled behavior;
7) realizing the production system on a computer;
8) displaying the performance of the formulated production system in a graph showing its coverage and error;
9) altering the production system until it fits the actual trajectory of the subject's search in the problem space;
10) testing the validity of production systems as "micro-theories" over a range of different subjects and/or tasks.[5]

In this paper, we have described the basic task environment (interactive composition with the POD program), diagrammed the basic problem space (Fig. 1), shown an actual protocol record (Table 1), inferred the augmented problem space (that of creating a satisfactory sonic repertoire and syntactic field to display a set of spectral envelopes), and described the nature of some specific sub-tasks. However, to give such an analysis some validity reaching beyond the specific instance quoted here, a formal model of the protocol activity would need to be formulated and implemented as a computer program whose output could be compared to that of subjects' behavior. Such a formal representation has been given by Laske in Backus normal form (BNF) grammar, [6] showing knowledge states, operators, and ordered sets of productions.

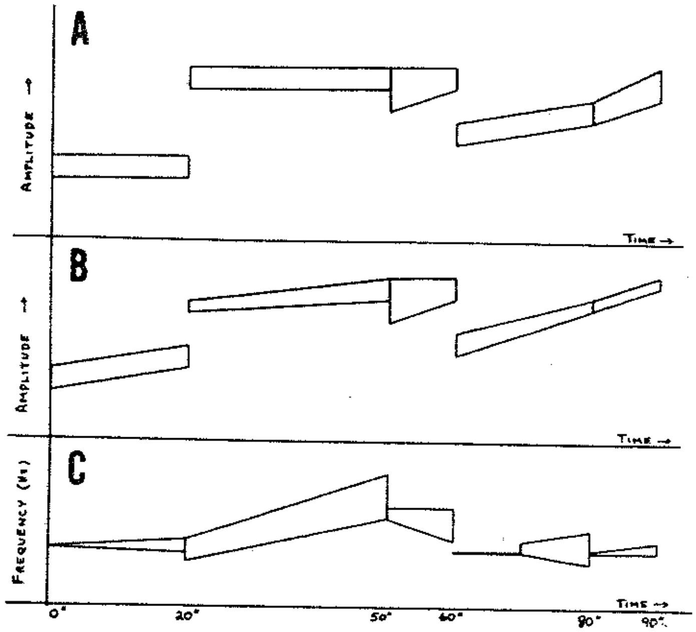

FIGURE 7. Protocol example: A) initial amplitude tendency mask; B) final amplitude mask; C) final frequency mask (modified from Fig. 6).

REFERENCES

1. Barry Truax, "The Computer Composition-Sound Synthesis Programs POD4, POD5, & POD6," *Sonological Reports* 2, Institute of Sonology, Utrecht, 1973; id., "Some Programs for Real-Time Synthesis and Composition," *Interface* 2 (1973): 159-62.
2. Truax, "Computer Music in Canada (1975)," pp. 22-23; id., "The POD Programs for Sound Synthesis and Composition at Simon Fraser University," unpublished manuscript, Vancouver, 1975.
3. John Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation," *Journal of the Audio Engineering Society* 21 (1973): 526-34. Reprinted in *Computer Music Journal*, 1(2):46-54, 1977.
4. Diana Deutsch, "The Organization of Short-Term Memory for a Single Acoustic Attribute," in *Short-Term Memory*, ed. D. Deutsch and J. A. Deutsch (New York: Academic Press, 1975), pp. 108-53.
5. Laske, *Information Processing Psychology Today*, p. 14.
6. Otto E. Laske, "Musical Acoustics (Sonology): A Questionable Science Reconsidered," *Numus-West* 6 (1974): 35-40.