Robertson Software, Inc.
1001 Pine Street, Boulder, CO 80302
303-443-5349 daver@robsoft.com

November 12, 1996

Mr. Bill Bettenberg
Department of the Interior
Office of Policy and Analysis
1849 C St. NW, Mail Stop 4412
Washington, DC 20240

Dear Mr. Bettenberg:

Having been contacted by Bill Sikonia about the further efforts to document and validate the TROA model, I am at his request writing you with my impressions of the issue along with some very personal thoughts on how best to proceed in the future. As you may recall, Kenn Cartier and I spent a couple of years in a vain attempt to document and rationalize the model. It was one of the more frustrating endeavors of my long software career. This career, now near a close, encompasses over 30 years of writing software.

It would be very easy (and tempting) to simply say that the model is "bad", should be thrown out forthwith, and a new model using modern software techniques be written to replace it. That perhaps begs the immediate question of whether the present model could stand up in court. My response to that question is twofold. First, no mere human could in good conscience profess to understand it in detail and prove that the results are "right". It is far too complex, convoluted, and baroque for that. On the other hand, it is probably difficult to prove that it is "wrong" in any overall sense, even though it is not hard to point out specific instances of inordinate sensitivity, small errors, arbitrary assumptions, and other problems.

It is difficult to separate the model from its long-time author, Rod Hall. I have come to respect Rod and his skill, as well as his ability to continue to work with this model with any degree of understanding. I also have the strong feeling that

EXHIBIT
TCID 159

there has been no attempt to fraudulently hide a hidden agenda within the ample recesses of the model.

The issue of starting over from scratch is a familiar one in computing circles. Rarely have I seen a case where the need to start over is more clear-cut. The TROA model, as I will delineate below, cries out for a new start when viewed from any number of different perspectives. This need must be balanced against another truth in software circles. New software always takes vastly longer and costs vastly more to create than anyone is willing to admit at the outset. To summarize, it needs to be rewritten, but it will be a painful, slow process.

My own feelings, arrived at from working with and upon the model for some years, are listed below.

1) The model is hopelessly complex for anyone to understand. This stems from multiple causes:

   - It is accretionary in that code has been added and added with no attempt to root out obsolete portions or consolidate similar functions. This, coupled with the lack of a crisp overall strategy, antique coding practices, and dubious shortcuts, makes it impossible to defend.

   - There is not adequate internal documentation..

   - It has been laid out with an enormous number of software "switches" to allow different paths through the program. Many of these switches are interrelated in unspecified ways.

   - The program is so large, has so many variables, constants, and input parameters that computational side effects are inevitable.

2) Detailed internal documentation is most apt to document that complexity without making the model any easier to understand. Documenting the overall strategy is probably impossible given the lack of an overall design strategy. At least I find it in general beyond the level of complexity I can understand, documented or not.

3) Our work showed various sensitivity problems where tiny differences caused substantial differences in paths through the code.

4) We did not find egregious errors that caused major differences in results, but then we were mainly trying to modernize the code without changing the essence of the computations.

Here are a few of my thoughts on appropriate paths for the future.

1) I don't think Rod could do the internal documentation effectively himself, as he has had no history of even attempting it. Further, he is burdened by a great deal of knowledge which would be counter-productive in deciding what to spell out and what to assume when composing comments..

2) Rewriting from scratch will be, at the least, a multi-year project, even assuming a good team of several people. It could be much longer. A new simplified model would be stalled over and over again as the political side wrangled over how to simplify. There are a lot of oxen being protected, as well as a changing real situation.

3) A new model should start from a viable commercial modeling platform such as Hydrosphere's CRAM platform. A second basic requirement must be a clear understanding of the role of the computer model. Is it for operating use, negotiation use, or some other use?

4) As the TROA world regards the present model much as a religion, no matter how good a new model is, it will be measured against the old TROA "King James" version with the burden of proof resting on the new, not the old. It will seem a major hurdle to shift this burden without demonstrating each error in the old.

I am including a few excerpts from e-mail correspondence between Kenn Cartier and myself as we labored to document and understand the model. Be aware that these are contemporaneous candid comments between colleagues, not carefully crafted public memos. I include them to document the difficulties in working

with the model, and the nature of some of the sensitivity problems we encountered

<u>DR to KC:</u> flsprt is a real bastard, isn't it. It would be so much easier to sit down with Rod and let him talk while we typed comments. While that would use more of his time, the aggregate time of all of us would surely be cut by 2/3. Oh, well. I say this not to justify our slow progress, but rather to emphasize the difficulty of understanding undocumented, accretionary code hoary with age. This might even get us little more of Rod's time. Would must be careful to not malign Rod other than possibly to point out the long term benefits of heavy commenting at the time code is written.

<u>KC to DR:</u> Dear comrade on the NSM front line (or maybe rear guard) I still am fighting in the trenches over the flsprt routine. I'm attempting to understand both the details and the big picture and to modify the comments accordingly. It's been slow going though.

<u>DR to KC:</u> It is hard to believe, but all the differences stem from our old friend rtg2. Apparently very minor differences in calculation order give very slight differences in results, which then propagate to major differences in other parts of the run. I'm going to put some additional debug statements in rtg2 to see the magnitude and paths of the differences. If they are small, I think someone should make some serious sensitivity checks on the code in general. The 115 area of INSTRM seems suspect as a tiny difference in Rcal(5) and Rtar(5) cause a different route and subsequently significant differences in results -- apparently all from rtg2 being different by 1 af/mon.

<u>DR to KC:</u> After spending quite a bit of time looking at side effects of slightly different orders of calculation of polynomials in rtg2, I think we need Rod's insight on how to proceed.

Background: Old way of calculating YVAL is:

```
      YVAL = Ca(4, j)*XVAL
      k = 3
      DO 250 l = 1, 2
         YVAL = XVAL*(YVAL + Ca(k, j))
         k = k - 1
250   CONTINUE
      YVAL = YVAL + Ca(1, j)
```

New way is:

```
      YVAL = Ca(1,j) + XVAL * (Ca(2,j) + XVAL *
            (Ca(3,j) + XVAL * Ca(4,j)))
```

Same calculation, but different sequence of multiplies and adds. On my PC, at least, this sometimes gives differences in about the 7th or 8th digit. That is OK in itself, as it probably doesn't translate into more than a bucket or two of water. The effect we see, though, may be quite a bit larger. In particular, when I ran a 1981-1983 case, the EOM storage in 1/83 in Independence is 15.267. However, in SVINDG, at line 145 (snippet follows) Stor(5) differs by a bit or so in the two different cases, causing sto to be 14.750 in one case and 15.267 in the other as the test goes the other way.

This subsequently causes quite a few calculations to be substantially different as the code follows different paths.

While we can make the results come out the same by replicating the old rtg2 calculation, it seems to me that there may be bigger problem when tiny differences can propagate into large ones.

DR to KC: Well, this is really interesting and, even though it has been in court many times, I think we now know that it is a pretty sensitive program. I've made similar changes to other parts of the code so who knows what will crop up. Note that Rod made the distinction that different operation procedures are invoked in different cases, perhaps based on very small differences in the decision criterion. This leaves me in somewhat of a quandary relative to our future testing. It would seem difficult to really clean up without making ANY calculation cleanups. Yet if we do, the burden on us to show that one answer is as good as another would seem almost insurmountable in the time at our disposal.

I don't relish the prospect of turning this into a full-blown analysis of the model, but at the same time it would seem that we have something of a burden to point out to interested parties that results can occur at the whim of roundoff changes.

KC to DR: I've isolated the problem in florat.f. Once again, it is an example of extreme sensitivity. Rod's working routine has the following command lines:

```
bomsto  = Stor(1)  -  Tcuic(1) - Tmic(1) - Clcrd(1)
eomsto  = Stor2(1) -  Tcuic(1) - Tmic(1) - Clcrd(1)
```

The discrepancy with our cleaned routine occurs because we altered the command lines to:

```
bomsto  = Stor(1)  - (Tcuic(1) + Tmic(1) + Clcrd(1))
eomsto  = Stor2(1) - (Tcuic(1) + Tmic(1) + Clcrd(1))
```

Although our lines are mathematically equivalent to Rod's command lines, the modification produces substantial differences in the output results.

DR to KC: I think we should be very very careful about making any quality or error comments to anyone other than Tom and Rod. We should only let anything go public, and by that I mean even to others in our own immediate groups, after discussion and agreement on the issue with Rod. We need to keep him on our side and also I don't think we wish to get in the position of being political footballs for those with different agendas to kick around. If we do come across a significant error, we should advise Rod, and if it needs to become public, let him do it in his own way. Guess that makes us mother's little helpers, but that is probably the best way.

I probably shouldn't have broadcast my 'Stange Code' e-mail to Yardas for that reason. I do think it is perfectly OK to bemoan the '60s style of the program, its lack of comments, and general inpenetratability. These are just my thoughts about how we should act, not a demand that we do so.

As a final summary, the points I would like to make are that the present model is not understandable, and probably not defensible at any deep level. It needs to be done over with a clear goal in mind and a modern software platform underneath. Such an effort will be costly and time consuming.

Truly yours

David Robertson
President
Robertson Software, Inc.

cc: Bill Sikonia
Bill Greer
Fred Disharoon
Jeff Zippen
Larry Bohman
Jon Nowlin
Kenn Cartier
Rod Hall