Part 3 Issues (Mostly)

1. Need to finalize the functional requirements.
2. Section 7.2.2, General Functional Requirements, requirement 1: A requirement on the validator doesn't seem to be a functional requirement.
3. Documentation requirements in Section 7 don't seem to be functional requirements. Have a separate section on documentation requirements?

   Look at all the documentation requirements. Are they appropriate for DRBGs? Do they belong in Section 7 at all?
4. Section 7.2.5, Internal State: Does the distinction between the various information in the internal state need to be made (i.e., administrative portion vs. working portion)? This seems to serve no useful purpose (at present), so just seems to make the state more complicated.
5. Section 7.2.7, Output Generation Function, requirement 3: Should we require the state to change whenever output is produced, or can we just require that the same bits from the internal state shall not be reused?
6. Section 8.3, para. 2, last sentence: Should we really require that the other input be unguessable? We're no longer providing additional input. Other input is now the personalization string (part of the seed construction) and other DRBG initialization stuff.
7. Section 8.4, Seeds, item 6, para. 3, 2nd sentence: Should prediction resistance be mentioned here?
8. Section 8.7, Backtracking and Prediction Resistance: How do I get the indistinguishability concept in here?
9. We need to thoroughly discuss the conceptual API in Part 1. What needs to be included there, and what is really internal to the RBG?
10. Section 9.6.1: Does the instantiation call for a DRBG really need a full entropy flag?
11. Section 9.6.1: Can the requested strength parameter be left out if an implementation only supports one security level? An application would need to be aware of the DRBGs implementation so as not to assume a greater strength than the DRBG can provide?

    The same question arises for other parameters (e.g., usage class, prediction resistance flag, personalization string, etc.)
12. Section 9.6.2, Request for Entropy: Should the *min_entropy*, *min_length* and *max_length* parameters be allowed to be missing when an implementation is intended to always use a single value? Can the *min_length* and *max_length* parameters be combined when they are intended to be the same? See the para. under the numbered list.
13. Section 9.6.4.3: What do we want to use as the block cipher derivation function? We are currently specifying the AES key wrap for both TDEA and AES.

14. Section 9.7, Reseeding: Does the conceptual API in Part 12 need a reseeding function? Should teh usage class be allowed to be omitted when an implementation only has one usage class?

15. Section 9.8, Generating Bits: The order of the input parameters is different in Part 1. Which should change?

16. Section 9.8: Part 1 needs to add an optional additional input parameter. Does Part 3 need to handle the full entropy flag?

    Can some of the input parameters be omitted in an implementation always handles only a single value for the parameter?

17. Section 9.9, Inserting additional entropy between requests: Do we want to include this process?

    Para. 1, last sentence: Should this be incorporated (i.e., if insufficient entropy is available, update anyway or don't update).

    Can some input parameters be omitted?

18. Section 10.1.2.1, Table 1: Are the security strengths in this table correct? Are the seed length ranges correct?

    Need to correct the DRBG to provide backtracking resistance.

    Need to provide more guidance on values for *t* for various applications.

19. Section 10.1.2.3.2: Should the seedlen be max (strength + 64, outlen) or be fixed for each hash function (e.g., SHA-1 = 192, SHA-224 = 256, SHA-256 = 320, SHA-384 = 384, SHA-512 = 512)?

20. Hash_DRBG: Are the following processes handled appropriately?

    Instantiation, including use of the personalization string, use of the derivation function, and transforming the seed material?

    Reseeding, including combining the old entropy with the new entropy bits.

    Generating random bits, including the handling of the prediction resistance flag, updating when the maximum number of updates is reached.

    Adding entropy to the Hash_DRBG: see issue 17 above.

    Should discussions be included as to which steps can be omitted under what conditions (currently included in the specs.)?

    What needs to be discussed re the generator strength and attributes? What is the suggested reseeding limit (this would be the value of *max_updates*)?