OO Design Review Guidelines

OBJECT-ORIENTED ANALYSIS - DOCUMENT CRITERIA

Spencer Rugaber

Copyright 1999, Spencer Rugaber

Version 1.9 November 8, 1999

Legend:

(c) for items that should be checked before the actual review;

(d) for items that should be part of the detail design review;

I. OBJECT CLASSES

- Does the class denote a collection of similar instances?
- If not, has the possibility of using a singleton been documented?
- Have object responsibilities been described adequately?

(c) Are objects annotated to indicate whether they are internal or external?

II. ATTRIBUTES

(c) Does each attribute have an associated data type?

(c) Are all data types primitives?

- If the data type is not primitive, should it be replaced by an association to an existing class?

(d) Have initial values been specified for attributes?

(d) Has a pointer attribute been used in place of an association?

(c) Have state invariants been expressed?

(c) Have derived attributes been indicated?

(c) Have class attributes been indicated?

III. OPERATIONS/SERVICES

(d) Are the arguments/results of the operation specified (names, types, mode (base/derived))?

- For a polymorphic operation, is it specified with a virtual function definition at the appropriate place in the class hierarchy?

(d) If the operation is polymorphic, is its signature consistent?

(c) Are operations annotated to indicate whether they alter the state or merely examine (and possibly return) it?

(d) Are operation preconditions indicated?

- Is the function/responsibility of the operation indicated?

(d) Are operation state alterations/postconditions described?

(c) Are any of the operations restrictions on operations of a parent class?

(c) Have class operations been indicated?

(c) Are operations annotated as to whether they are actions or activities?

(c) If an operation in a class overrides an operation in the parent class,
    is the return type of the overriding function a subtype of the return
    type of the overridden function (covariance)?
(c) If an operation in a class overrides an operation in the parent class,
    is the type of each parameter of the the overriding function a supertype
    of the type of the parameter in the overridden function (contravariance)?

IV. ASSOCIATIONS

(c) Is the arity of each association indicated?
 - Can associations with arities > 2 be replaced by binary associations
    meaningfully?
(c) Are cardinality constraints indicated?
(d) Is any symmetry of each association indicated?
(c) Are role names given for each of the classes involved in a recursive
    association?
(c) Have any ordering constraints been indicated?
 - Have qualifications been provided to reduced the multiplicity of
    associations?
 - Can a one-to-one association be made multiple?
(c) Have any referential integrity constraints been indicated?
(c) Does each association require a persistent representation? If not,
    could it be better modeled as an operation?
 - Should the association be normalized by introducing intermediate classes
    and associations?
 - Have link attributes and operations been annotated?

V. AGGREGATION ASSOCIATIONS

 - Could an aggregate be better modeled with attributes?
(c) Is the association transitive?
(c) Is the association anti-symmetric?
(c) Have existence dependencies been indicated?
(c) Are propagation properties indicated?

VII. GENERALIZATION ASSOCIATIONS

(c) Have common attributes of several classes been factored into a superclass?
(c) Is the association transitive?
(c) Have common methods been moved to the superclass?
(c) Have discriminators been indicated for each generalization?
(c) Have restrictive overrides been annotated?
(c) If a set of sibling subclasses differ only in the value of one
    attribute, could the situation be better modeled by an (enumerated)
    attribute in the parent class?
 - Have overlapping memberships been indicated for non-disjoint subclasses?
 - Does the set of subclasses completely partition the parent class?

VIII. SCENARIOS / EVENT TRACES

(c) Are architectural scenarios separated from detail scenarios?

(c) Are message cascades sufficiently abbreviated?

- Have transitions been checked for the necessity of guards?

- Have cascaded events been checked for race conditions?

IX. INTERMODEL CONSISTENCY

(O indicates object model; D, dynamic model; and F, functional model)

(c) Are data conditions (triggers) indicated? (O-D)

- Do they cause appropriate events?

(c) Are actions (operation invocations) indicated in the control model? (D-F)

(c) Are events indicated in the functional model? (F-D)

(c) Are all message names from scenarios defined as object operations?

X. ARCHITECTURAL CONSIDERATIONS

(c) Have all three views of the problem space been provided (data, function, control)?

- Has the class hierarchy been examined for reuse possibilities?

- Have potential frameworks been identified and documented?

- Does the documentation include a description of the collaboration pattern (intra-framework protocol) for the framework?

- Have the system boundaries been specified?

XI. USE OF NAMES

(c) Does the data dictionary exist?

(c) Are all names used in models and diagrams included in the dictionary?

(c) Is each name unique?

(c) Does each data dictionary entry have associated descriptive commentary?

- Is the entry's name descriptive of its role?

(c) Are naming standards conformed to?

XII. GENERAL DOCUMENT CHECKS

(c) Does the document conform to style standards?

(c) Does the document include a glossary?

(c) Does the document include a table of contents?

(c) Does the document include an index?

- Does the document present a uniform level of detail?

(c) Are all diagram conventions adhered to:

labels

balancing

XIII. REQUIREMENTS TRACING

- Has the requirements document been examined to determine if all
  system constraints have been indicated as associations or been
  assigned to operations to satisfy?
- Has the requirements document been surveyed for objects, attributes, and
  methods (nouns, adjectives, and verbs)?

(c) Are model features annotated with relevant requirement numbers?

- Have all requirements been addressed by one of the models?

XIV. STRUCTURAL CHECKS

- Suggest five plausible modifications to the requirement. Evaluate
  which objects are affected by each change. Judge the value of the
  model by the localization of the changes.
- Select five likely scenarios. Create the corresponding interaction
  diagrams. Look at message traffic patterns. Detect situations where
  responsibility for a relatively small task is shared among several
  objects, resulting in heavy bilateral traffic.
- Have a variety of viewpoints been examined (e. g. services, administration,
  debugging)?