Model Review Checklist

Work Product(s) under Review	
Product / System name	
Product/Name ID	
Configuration ID	
Product type	**Model**
Product(s) Owner	
Reviewer(s)	
Review date	
Standard(s) applied	

Note: This checklist applies to various kinds of models and viewpoints. Not all viewpoints may apply to the model or model aspects under review.

Note: For the purpose of this document, the term "Class Diagram" refers to all named uses of a class diagram, including class, object, block, structured, task, and subsystem diagrams.

	Approved Y/N	**Item**	**Required actions / Comments**
General Modeling Guidelines			
☐		Is the model organized along the project model organization structure?	
☐		Are requirements located entirely within one high-level package?	
☐		Are use cases and related functional analysis model elements located within one high-level package?	
☐		Are architectural views located within one high-level package?	
☐		Are the subsystems located within separate projects (models) including their specific requirements, use cases, and internal design?	
☐		Are (internal) model elements shared across multiple teams located in a *Domains* package within a commonly shared model?	
☐		Are (external) model elements used in interfaces between subsystems or components located within an *Interfaces* or *Common Types* package within a commonly shared model?	
☐		Is the Domains package used by reference in the subsystem model?	
☐		Is the Domains package in the shared model organized by subpackages, each of which is specific to a subject matter area?	
☐		Are domain subpackages related by at most a single usage dependency from most abstract towards least abstract (i.e. no circular or backwards dependencies)?	
☐		Are normative and non-normative model elements clearly identified?	

	Approved Y/N	Item	Required actions / Comments
☐		Do comments clearly establish *why* something is depicted, modeled, or represented?	
☐		Are special “to do” comments clearly identified with push pin icon or some other visual element that distinguishes them from persistent explanatory comments?	
☐		Are diagrams connected to each other with navigable hyperlinks that facilitate model navigation?	
☐		Are important model elements connect to important related diagrams via navigable hyperlinks?	
☐		Do model elements have traceable links to the requirements they represent or realize, including:	
☐		- classes	
☐		- use cases	
☐		Does each model element have a description that includes - Purpose - Description - Preconditions - Postconditions - Invariants including the following kinds of elements	
☐		- package	
☐		- use case	
☐		- class	
☐		- method, operation, function	
☐		- data type	
☐		- variable	
☐		- diagrams, including	
☐		- use case diagram	
☐		- class diagram	
☐		- sequence or communication diagram	
☐		- activity diagram	
☐		- timing diagram	
Naming Conventions			
☐		Are model element names taken from appropriate *problem domain* vocabulary?	
☐		Do model names avoid the use of white space and special non-alphanumeric	

	Approved Y/N	Item	Required actions / Comments
		characters?	
☐		Do class names start with upper case letters?	
☐		Do use case names start with upper case letters?	
☐		Do event names begin with upper case letters?	
☐		Where appropriate, are model element names “compilable” for the intended target language?	
☐		Do class or use case “features” start with a lower-case letter, including	
☐		- attributes	
☐		- operation, event receptors or method names	
☐		- ports	
☐		- parts (e.g. contextualized object roles within a composite classifier)	
☐		- association role names	
		Do object names begin with lower case letters?	
☐		Are structural elements named with a strong noun or noun phrase, including	
☐		- classes	
☐		- objects	
☐		- attributes	
☐		Are behavioral elements named with a strong verb or verb phrase, including	
☐		- operations and methods	
☐		- event receptors	
☐		- use cases	
☐		- messages (e.g. on sequence diagrams)	
☐		Are states named according to “conditions of existence” or modes of operation?	
☐		In multi-word names, is each word (after the first) begun with an upper case letter?	
☐		Do interface names between with the prefix “i” (lower case “i”)?	
Activity Diagrams			
☐		Is the activity diagram model primarily flow-of-control algorithms rather than state-based flow?	

	Approved Y/N	Item	Required actions / Comments
☐		When activities within an activity diagram are allocated to different elements, are that indicated with swim lanes?	
Class diagrams			
☐		Does each class diagram have a singular mission which is explicitly stated on the diagram and followed?	
☐		Does each class diagram have all elements relevant to its stated mission?	
☐		Does each class diagram elide all elements not relevant to its stated mission?	
☐		Are all classes in the model represented in at least one diagram?	
☐		Are all associations, aggregations, compositions, and generalizations among the classes represented on at least one diagram?	
☐		Does the model contain at least one class diagram addressing each of the following missions?	
☐		- Subsystem or component architecture	
☐		- Concurrency and resource architecture	
☐		- distribution architecture	
☐		- safety, reliability, and security architecture	
☐		- deployment architecture	
☐		- realization of each realized use case	
☐		- structure of a composite class	
☐		Does the diagram minimize “line cross” of represented relations?	
☐		Are the relation types shown with consistent line styles, including	
☐		- generalization (straight)	
☐		- association (rectilinear)	
☐		- aggregation (rectilinear)	
☐		- composition (rectilinear)	
☐		- dependency (rectilinear)	
☐		Where subclassing is used, is Liskov substitutability maintained?	
☐		If subclassing is used, is it truly a case of subclasses and not instance usage?	

	Approved Y/N	Item	Required actions / Comments
☐		Are the font styles and sizes consistent within the diagram?	
☐		Are classes similarly sized?	
☐		- for canonical form (no compartments)	
☐		- for compartment form	
☐		Are composite structured classes sized appropriately to show their internal parts relevant to this mission of the diagram?	
☐		Is the use of color to show semantic information avoided?	
☐		Are role names on associations, aggregations and compositions shown on all role ends?	
☐		Is multiplicity shown on all role ends for associations, aggregations, and compositions (note: only required for “part” end of composition)?	
☐		Are all associations depicted as bi-directional truly bi-directional?	
Model Overview Diagram			
☐		Does the model have a Model Overview Diagram contains links to relevant model entry points and diagrams?	
☐		Are there hyperlinks to model entry points including	
☐		- Requirements	
☐		- Use case analysis	
☐		- Architecture	
☐		- Design	
☐		- Dependability analysis (i.e. safety, reliability, and security)?	
☐		- Performance and tradeoff analysis	
☐		- Data and type definitions	
☐		- Tests	
☐		- Interfaces	
☐		Does the Model Overview Diagram clearly state the purpose and scope of the model?	
Sequence Diagrams			
☐		Does each sequence diagram have a comment that identifies its purpose and scope, including: - Purpose (for example + black box use case scenario	

	Approved Y/N	Item	Required actions / Comments
		+ white box use case scenario + “part” scenario extracted from a larger one + design scenario depicting interaction of design elements + animated scenario, capturing the execution of system elements + test case specification) - Description - Preconditions - Postconditions - Invariants	
☐		Are the lifelines for each sequence diagram appropriate to its purpose (e.g. if the SD is a black box use case scenario, lifelines should only be actors and the use case)?	
☐		Is each sequence diagram commented adequately to enable understanding of the important aspects of the control and/or data flow?	
☐		Does each message of the sequence diagram include parameter data type and (where appropriate) range or value?	
☐		Is each message properly shown as synchronous or asynchronous?	
☐		If a lifeline on a sequence diagram is decomposed, do the messages into and out of that lifeline at the higher-level abstraction SD match the messages at the lower level abstraction?	
☐		Are interaction fragments limited to at most three levels of nesting?	
State Diagrams			
☐		Is the default state for every level of nesting identified with a default pseudostate?	
☐		Is the state machine compilable and executable?	
☐		Are the names of the states reflective of appropriate problem domain vocabulary?	
☐		Are and-states used if and only if the order of execution of actions in different and-	

	Approved Y/N	Item	Required actions / Comments
		states irrelevant or largely so?	
☐		Where and-states are used, is the system free of race conditions?	
☐		Do guards lack side effects?	
☐		When multiple guards exist exiting a conditional connector, do they specify non-overlapping conditions?	
☐		Are null-triggered and anonymous states only used to appropriate force state machine closure?	
☐		Are complex action lists clustered into local operation/method calls?	
☐		Are submachines used to appropriately manage nested state complexity?	
☐			
☐			
Use Case Diagrams and Use Cases			
☐		Does each use case represent a system capability or usage?	
☐		Does each use case have traceable links to requirements (> 5) that it represents?	
☐		Is each use case independent *in terms of requirements* from the other use cases (i.e. each requirement binds to at most one use case)?	
☐		Does each functional and quality of service requirement map to one use case?	
☐		Is the use case named with a strong verb or verb phrase?	
☐		Does each use case associate with at least one actor?	
☐		Is each use case elaborated with 3 or more sequence diagrams, each of which has more than 3 messages?	
☐		Is each use case specified by a normative state machine?	
☐		Does each use case have a description with the format - Purpose - Description - Preconditions - Postconditions - Invariants	

	Approved Y/N	Item	Required actions / Comments
☐		Does each use case diagram have a comment that describes is scope and content?	
☐		For use case scenarios, are all important variants of the scenario represented in the set of associated sequence diagrams?	
☐		Is every requirement bound to the use case bound to at least one element on one sequence diagram?	
☐		Are both normal and exception and error handling cases represented in the set of scenarios for the use case?	
☐		Is every message and action on the set of sequence diagrams elaborating a use case represented in that use case's state machine?	
☐		Is every transition and action on a use case state machine represented in at least one use case sequence diagram?	
☐		Is every actor a physical or logical entity outside the scope of the system development?	
☐		Does the use case hide "how" the behavior will be achieve and instead focus on the required data and control transformations that are required?	
☐		Does the use case model hide any interaction of the actors as "out-of-scope"?	
☐		Are time-based transitions represented NOT as an actor, but instead represented as internal events on the sequence diagrams and state machine?	
☐		If use cases are specialized, is the use of generalization / specialization appropriate?	
☐		If large use cases are decomposed, is the <<include>> relation used appropriately?	
☐		Is the use of associations between use cases avoided?	
☐		Is the state machine of the use case executable (alternatively – a block/class diagram representing the use case can be constructed and, if so, is THAT state machine executable)?	

	Approved Y/N	Item	Required actions / Comments
☐		Are the logical interfaces between the actor and the system (executing the use case) clearly identified in terms of messages and passed data?	
Traceability			
☐		Do model elements have traceable links to the requirements they represent, elaborate, specialize or realize, including:	
☐		- classes	
☐		- use cases	
☐		- stand lone functions	
☐		- data types	
☐		- variables	
☐		- interfaces	
☐		- class features	
☐		- attributes	
☐		- operations or methods	
☐		- event receptors	
☐		- ports	
☐		- state machines	
☐		- states	
☐		- transitions	
☐		- state machine actions	
☐		- associations, aggregations, and compositions	
☐		Do model elements have traceable links to the test cases, including:	
☐		- classes	
☐		- use cases	
☐		- stand lone functions	
☐		- data types	
☐		- variables	
☐		- interfaces	
☐		- class features	
☐		- attributes	
☐		- operations or methods	
☐		- event receptors	
☐		- ports	
☐		- state machines	
☐		- states	
☐		- transitions	
☐		- state machine actions	

	Approved Y/N	Item	Required actions / Comments
☐		- associations, aggregations, and compositions	