Département de génie informatique et de génie logiciel

LOG3430
Méthodes de tests et validations

TP4
Tests basés sur les états

Soumis par :
Jean-Frédéric Fontaine (1856632)
Sébastien Cadorette (1734603)

15 octobre 2018
Soumis à : Hiba Bagane

Polytechnique Montréal

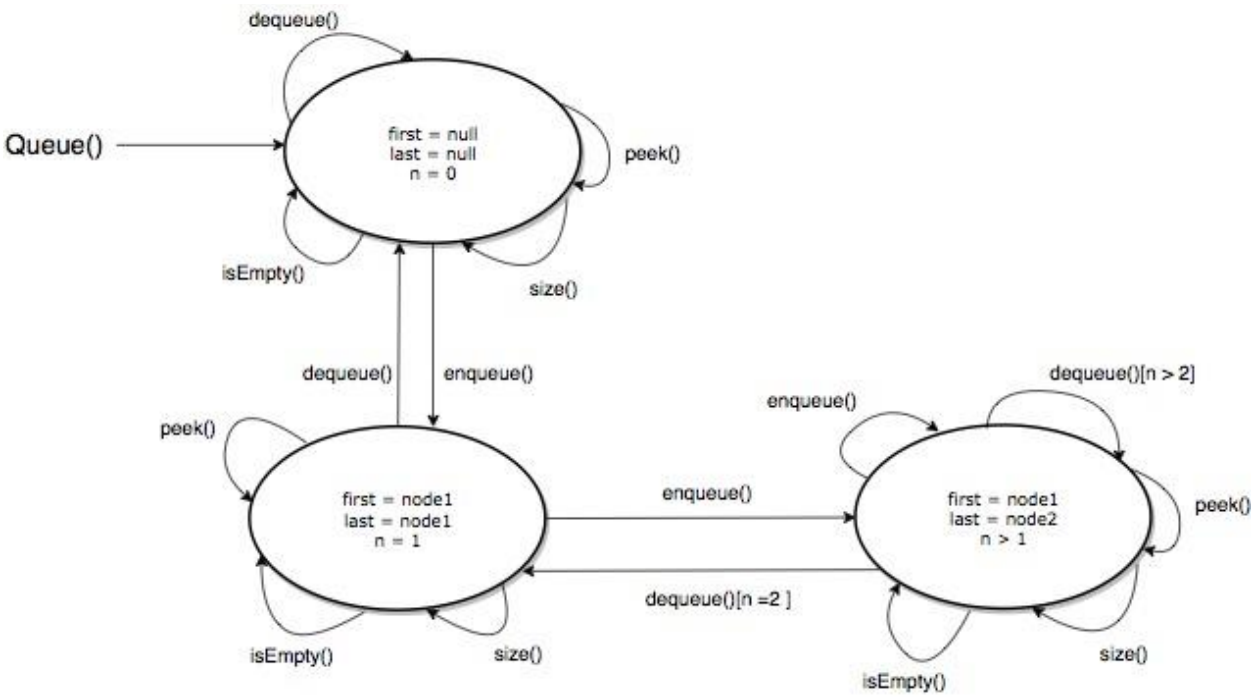
Table des matières

Diagramme d'états ***3***

Arbre de transitions ***4***

Séquences ***5***

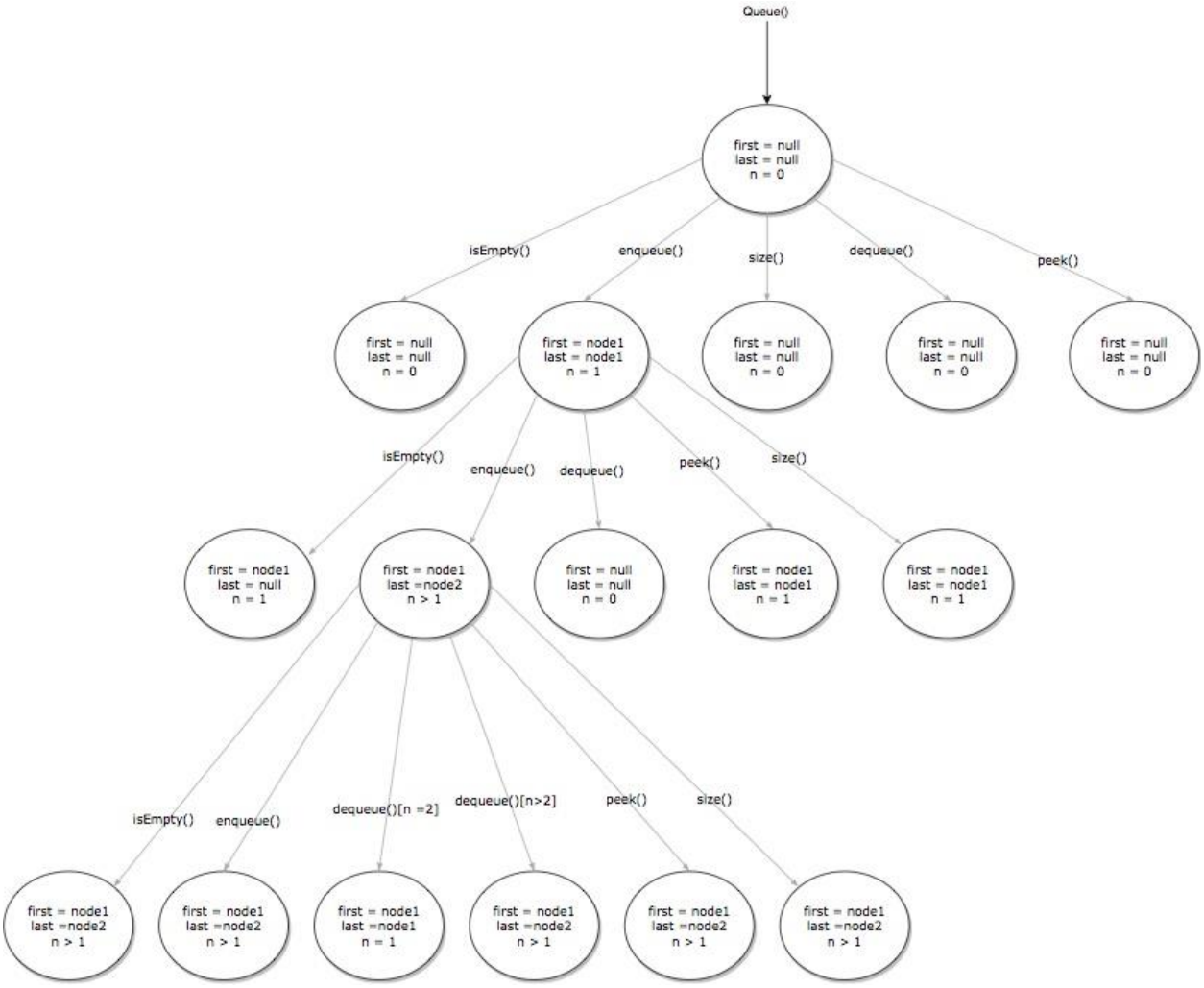
Diagramme d'états



Vous trouverez une version PDF de ce diagramme dans notre dossier de remise. Le format PDF étant plus adapté à la lisibilité de ce genre de diagramme, il sera plus clair de voir chacun des états.

Tel que mentionné dans l'énoncé, nous retrouvons 3 états. Le premier état se retrouve en haut complétement. L'attribut *first* et l'attribut *last* sont de valeur égale à *null*. L'attribut *n* est de valeur égale à 0. Le deuxième état est celui juste en bas du premier. L'attribut *first* et *last* sont de même valeur. L'attribut *n* est de valeur égale à 1. C'est l'état où la *Queue* ne comporte qu'un seul élément. Le *first* est également le *last*, et vice-versa. Le troisième et dernier état est celui situé au bas à gauche dans le diagramme. L'attribut *first* et l'attribut *last* aura des valeurs différentes de *null* et ne seront pas forcément égaux l'un l'autre. L'attribut *n* est de valeur supérieure à 1.

Arbre de transitions



Tout comme pour le diagramme des états, l'arbre de transitions est également sous format PDF dans notre dossier de remise.

Séquences

Pour identifier les cas de tests, on prend les feuilles de l'arbre de transition et on associe un test à la séquence des méthodes appelées pour se rendre à cette feuille.
Voici ainsi nos 14 tests identifiés à partir des 14 feuilles de notre arbre de transition.

d01 = < { Queue() -> isEmpty() } , { first == null, last == null, n == 0 } >

d02 = < { Queue() -> size() } , { first == null, last == null, n == 0 } >

d03 = < { Queue() -> dequeue() } , { first == null, last == null, n == 0 } >

d04 = < { Queue() -> peek() } , { first == null, last == null, n == 0 } >

d05 = < { Queue() -> enqueue(1) -> isEmpty() } , { first == 1, last == 1, n == 1 } >

d06 = < { Queue() -> enqueue(1) -> size() } , { first == 1, last == 1, n == 1 } >

d07 = < { Queue() -> enqueue(1) -> peek() } , { first == 1, last == 1, n == 1 } >

d08 = < { Queue() -> enqueue(1) -> dequeue() } , { first == null, last == null, n == 0 } >

d09 = < { Queue() -> enqueue(1) -> enqueue(2) -> isEmpty() } ,
{ first == 1, last == 2, n == 2 } >

d10 = < { Queue() -> enqueue(1) -> enqueue(2) -> size() } ,
{ first == 1, last == 2, n == 2 } >

d11 = < { Queue() -> enqueue(1) -> enqueue(2) -> peek() } ,
{ first == 1, last == 2, n == 2 } >

d12 = < { Queue() -> enqueue(1) -> enqueue(2) -> enqueue(3) } ,
{ first == 1, last == 3, n == 3 } >

Condition n > 2

d13 = < { Queue() -> enqueue(1) -> enqueue(2) -> enqueue(3) -> dequeue() } ,
{ first == 2, last == 3, n == 2 } >

Condition n == 2

d14 = < { Queue() -> enqueue(1) -> enqueue(2) -> dequeue() } ,
{ first == 2, last == 2, n == 1 } >

Afin de pouvoir vérifier proprement l'état dans lequel se retrouve chacun de nos tests, nous avons ajouté une fonction accesseur pour l'attribut *last*. Tel que vu dans le dernier travail pratique, la classe *Queue* ne fournit aucun accesseur pour l'attribut *last*.