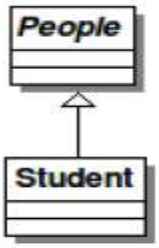
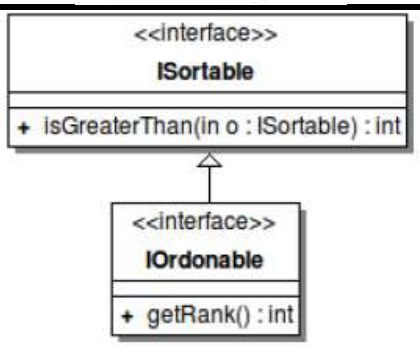
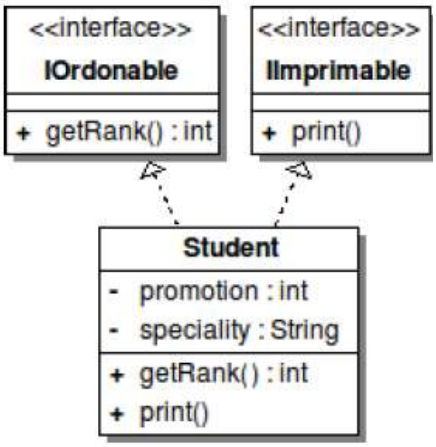
Correspondance UML-Java

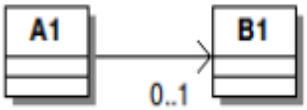
A. Diagramme de classes

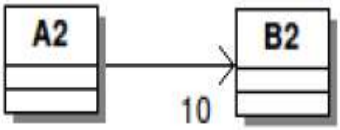
Type de relation	Représentation UML	Syntaxe Java

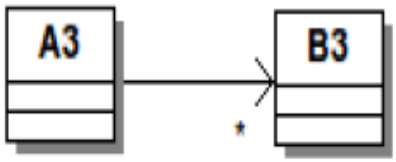
Héritage de classe		public abstract class People {
		…
		}
		public final class Student **extends** People {
		…
		}

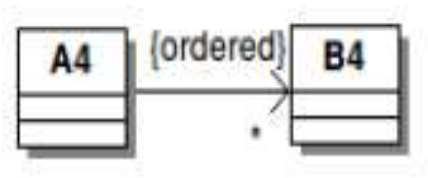
Héritage d'interface		public interface ISortable {
		public int isGreaterThan(ISortable o);
		}
		public interface IOrdonable extends ISortable {
		public int getRank();
		}

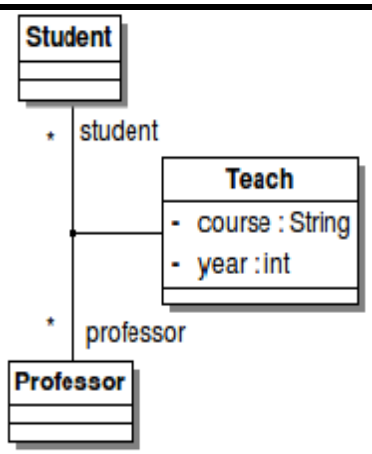
Réalisation d'interfaces		public interface IImprimable {
		public void print();
		}
		public interface IOrdonable {
		public int getRank();
		}
		class Student **implements** IOrdonable, IImprimable {
		private int _promotion;
		private String _speciality;
		public int getRank() {
		…
		}
		public void print() {
		…
		}
		}

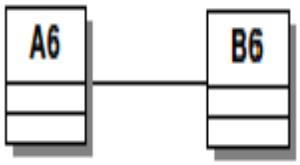
Association navigable de multiplicité 1		public class A1 {
		private **B1 b**;
		…
		}
		public class B1 {
		}

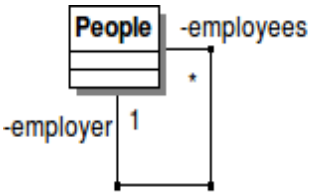
Association navigable avec une multiplicité fixée		public class A2 {
		private **B2[] bs = new B2[10];**
		…
		}

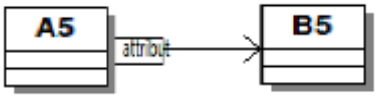
Association navigable avec une multiplicité quelconque		public class A3 {
		private **ArrayList<B3>** bs = new ArrayList<B3>();
		…
		}

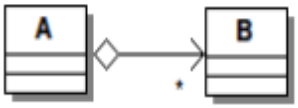
Association multiple ordonnées		public class A4 {
		private **TreeSet<B4> _b4s** = new TreeSet<B4>();
		…
		}
		public class B4 implements Comparable {
		int **compareTo**(B4 b) { }
		}

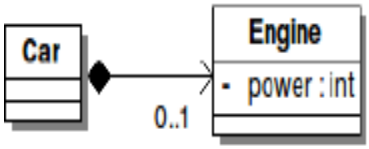
Association entre plusieurs classes		public final class Student {
		}
		public final class Professor {
		}
		public final **class Teach** {
		private string course;
		private int year;
		private **Professor prof**;
		private **Student std**;
		…
		}

Association sans navigabilité (bidirectionnelle)		public class A6 {
		private **B6 b**;
		}
		public class B6 {
		private **A6 a**;
		}

Association réflexive		public class People {
		private **ArrayList<People>** employees = new ArrayList<People>();
		private **People** employer;
		}

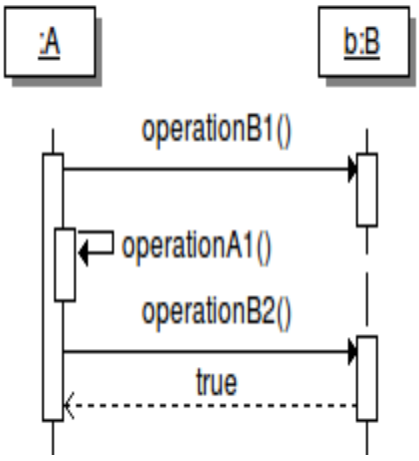
Association qualifiée		public class A5 {
		private **HashMap<attribut,B5> bs** = new HashMap<B5>();
		…
		}

Agrégation		public class A {
		private **ArrayList<B>** bs = new ArraList<B>();
		}

Composition		public class Car {
		private **Engine eng;**
		private Car() {
		eng = new Engine();
		}
		…
		}
		private class Engine {
		private int power;}

B. Diagramme de séquences

	Représentation UML	Syntaxe Java

Diagramme de séquences		public class A {
		private void operationA1() { … }
		public void scenario() {
		B b = new B() ;
		b.operationB1();
		this.operationA1();
		b.operationB2();
		}
		…
		}
		public class B {
		…
		public void operationB1() { … }
		public boolean operationB2() { … }
		…
		}