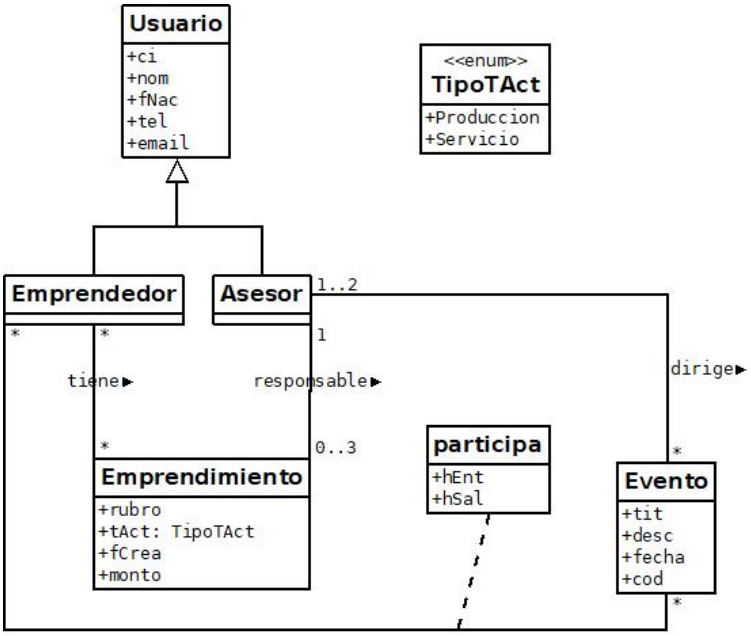
Programación 4

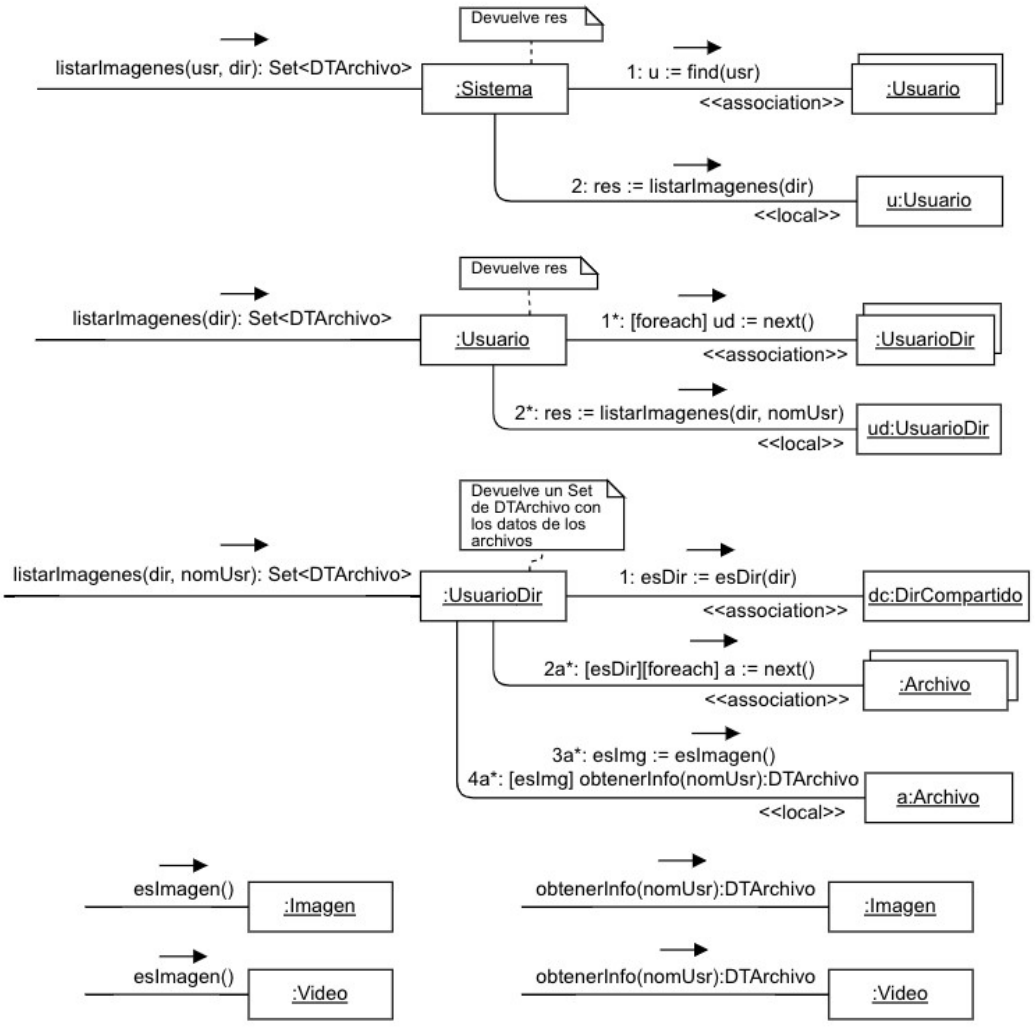
SOLUCIÓN EXAMEN DICIEMBRE 2020

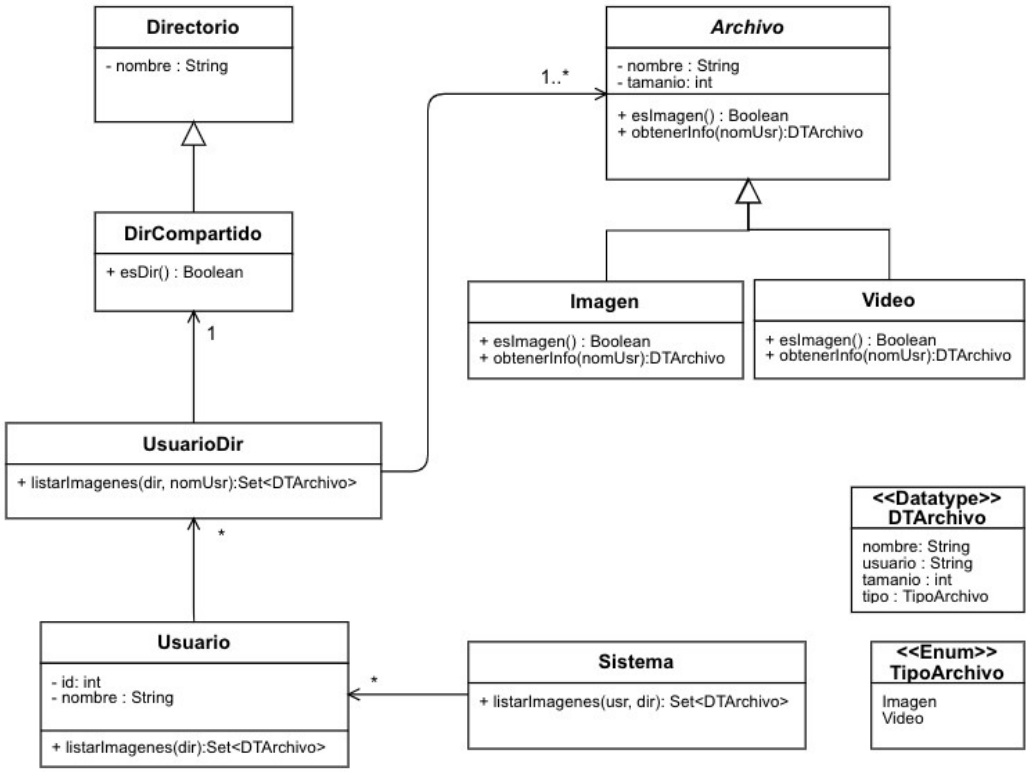
Problema 1 (30 puntos)



Aplica únicamente restricciones de identificador y dominio de atributos.

Problema 2 (35 puntos)




Directorio
- nombre : String
DirCompartido
+ esDir() : Boolean
1
UsuarioDir
+ listarImagenes(dir, nomUsr):Set<DTArchivo>
*
Usuario
- id: int
- nombre : String
+ listarImagenes(dir):Set<DTArchivo>
*
Sistema
+ listarImagenes(usr, dir): Set<DTArchivo>
1..*
Archivo
- nombre : String
- tamanio: int
+ esImagen() : Boolean
+ obtenerInfo(nomUsr):DTArchivo
Imagen
+ esImagen() : Boolean
+ obtenerInfo(nomUsr):DTArchivo
Video
+ esImagen() : Boolean
+ obtenerInfo(nomUsr):DTArchivo
<<Datatype>>
DTArchivo
nombre: String
usuario : String
tamanio : int
tipo : TipoArchivo
<<Enum>>
TipoArchivo
Imagen
Video

Problema 3 (35 puntos)

CtrlVentas.h

```
class CtrlVentas {
private:
	CtrlVentas();
	static CtrlVentas* instancia;
	map<string, Cliente*> clientes;
	map<LibroID, Libro*> libros;
public:
	bool comprarLibro(string ciCliente, LibroID idLibro);
	static CtrlVentas* darInstancia();
};
```

Libro.h

```
class Libro {
private:
	LibroID id;
	set<string> autores;
	float precio;
public:
	Libro(LibroID id, set<string> &autores, float precio);
	virtual bool hayDisponible(Cliente* c) = 0;
	virtual void disminuirStock() = 0;
	virtual void confirmarCompra(Cliente* c) = 0;
	virtual ~Libro() {};
};
```

LibroPapel.h

```
class LibroPapel: public Libro {
private:
	int cantStock;
	map<string, Cliente*> reservas;
public:
	LibroPapel(LibroID id, set<string> &autores, float precio, int cantStock);
	bool hayDisponible(Cliente* c);
	void disminuirStock();
	void confirmarCompra(Cliente* c);
	virtual ~LibroPapel() {};
};
```

CtrlVentas.cpp

```
CtrlVentas* CtrlVentas::instancia = NULL;

CtrlVentas* CtrlVentas::darInstancia(){
	if (instancia == NULL){
		instancia = new CtrlVentas();
	}
	return instancia;
}
```

```
bool CtrlVentas::comprarLibro(string ciCliente, LibroID idLibro) {
	Cliente* c = clientes[ciCliente];
	Libro*   l = libros[idLibro];
	if (l->hayDisponible(c)) {
		l->disminuirStock();
		l->confirmarCompra(c);
		return true;
	}
	else {
		return false;
	}
}
```

LibroPapel.cpp

```
LibroPapel::LibroPapel(LibroID id, set<string> &autores, float precio, int
cantStock)
	: Libro(id, autores, precio), cantStock(cantStock) {
}

bool LibroPapel::hayDisponible(Cliente* c) {
	bool reservado = reservas.find(c->getCi()) != reservas.end();
	bool hayStock  = (cantEnStock - reservas.count()) > 0;
	return reservado || hayStock;
}

void LibroPapel::disminuirStock() {
	cantEnStock--;
}

void LibroPapel::confirmarCompra(Cliente* c) {
	bool reservado = reservas.find(c->getCi()) != reservas.end();
	if (reservado) {
		reservas.erase(c->getCi());
	}
}
```