Interfaces

```
class ISimObject
{
	public:
	virtual string getName()=0;
	virtual int getXLoc()=0;
	virtual int getYLoc()=0;
	virtual int getType()=0;
};

class ISee
{
	public:
	virtual list<ISimObject *>& look( int x ) = 0;
};

class IProbe
{
	public:
	virtual list<ISimObject *>& probe() = 0;
};

class IMove
{
	public:
	virtual int getAction() = 0;
	virtual int getParameter() = 0;
};

class IRobot: public ISimObject
{
	public:
	virtual IMove & getMove( ISee & s, IProbe & p) = 0;
};

class ISimulation
{
	public:
	virtual void loadConfiguration( string file) = 0;
	virtual void setRobot( int index, IRobot & r) = 0;
	virtual void step( int s ) = 0;
};
```

Notes

As per recent requests, there are now XLoc and YLoc methods in ISimObject, as well as a getType.
For all objects passed in as pointers or references, you may only assume that those memory locations exist during that method call; therefore, you will need to duplicate anything you will need as persistent storage.

The integer returned by getType of ISimObject is one of the following:

0=Robot
1=Rock
2=Lava
3=Water
4=Mud
5=Ball
6=Block
7=Hole
8=EnergyPill
9=Fog
10=Jam

It should be noted that for this as well as for the Action set of possibilities, a good design will take into account future changes to this list, and code in such a way that changes would require minimal work.

It should also be noted that as per the spec, "The ISimObject represents any object or property in the simulation." and not just 'objects' as the term is used in the spec regularly.

If there are any extra methods that would be useful, please post to WebCT and we can discuss adding them to the spec. We will, however, strive for a minimal amount of required methods to allow for design decisions on the part of students.