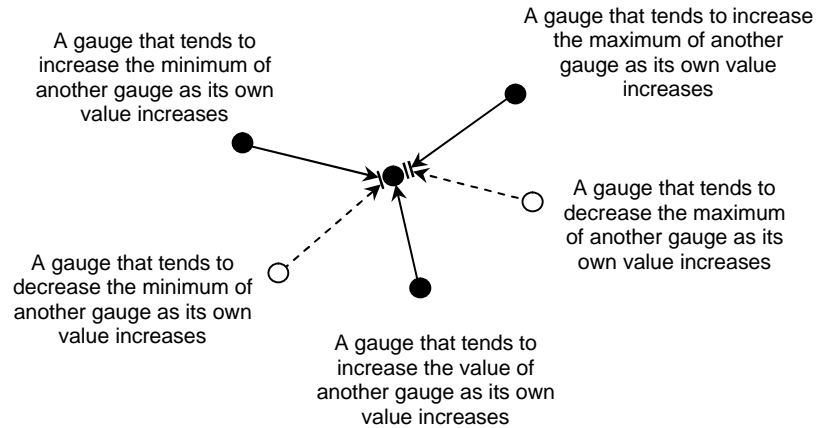# Proposed RPG Diagram Notation Changes

Revision 4

Gauge Diagrams

A gauge that tends to increase the minimum of another gauge as its own value increases

A gauge that tends to increase the maximum of another gauge as its own value increases

A gauge that tends to decrease the maximum of another gauge as its own value increases

A gauge that tends to decrease the minimum of another gauge as its own value increases

A gauge that tends to increase the value of another gauge as its own value increases
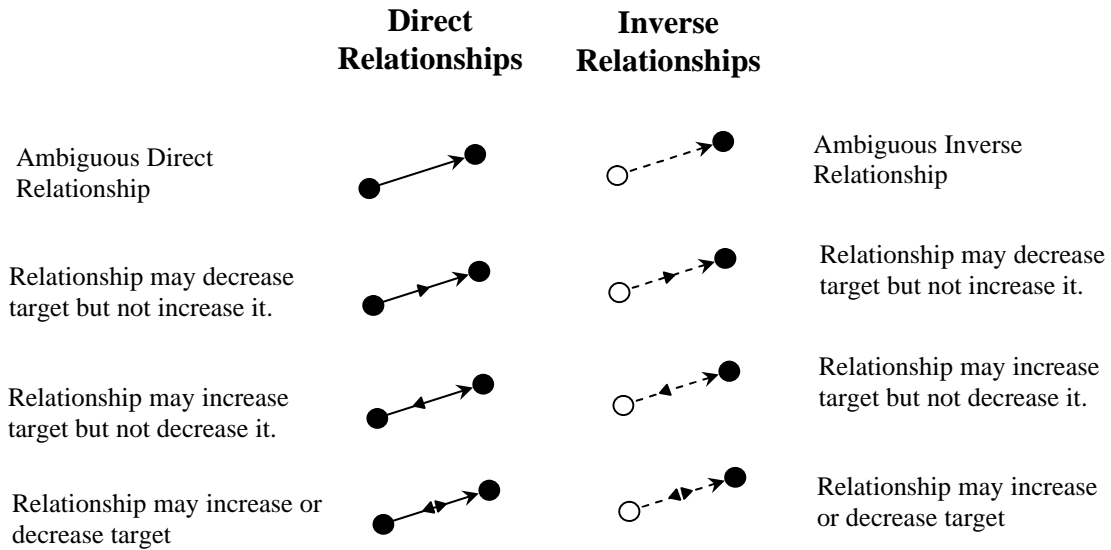
An "Element of" symbol: $\in$ can be used to identify a gauge as being an element of a set. In this way, relationships can specify whether they apply to individual set elements or to the set as a whole:

A gauge that tends to increase the maximum of all gauges in a set as its own value increases

A gauge that tends to decrease the maximum number of allowable characteristics in a set as its own value increases

A gauge that tends to increase the values of gauges in a set as its own value increases

A gauge that tends to increase the number of characteristics in a set as its own value increases

A gauge that tends to decrease the minimum of all gauges in a set as its own value increases

A gauge that tends to increase the minimum number of allowable characteristics in a set as its own value increases

If you want to distinguish the winning side from the losing side in a contest, fill in the winning side of the contest icon.

Winning Side

Losing Side

Success?

1
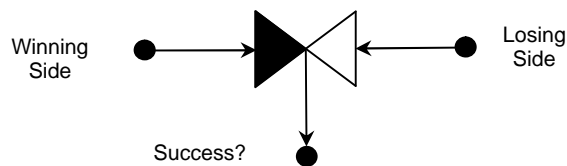
It would sometimes be useful to more clearly specify the precise nature of a relationship. Some relationships can increase, but not decrease the target. Some can decrease but not increase the target. Others can do both. To allow more detail, adornments can be added to provide this information:

Direct Relationships   **Inverse Relationships**

Ambiguous Direct Relationship

Ambiguous Inverse Relationship

Relationship may decrease target but not increase it.

Relationship may decrease target but not increase it.

Relationship may increase target but not decrease it.

Relationship may increase target but not decrease it.

Relationship may increase or decrease target

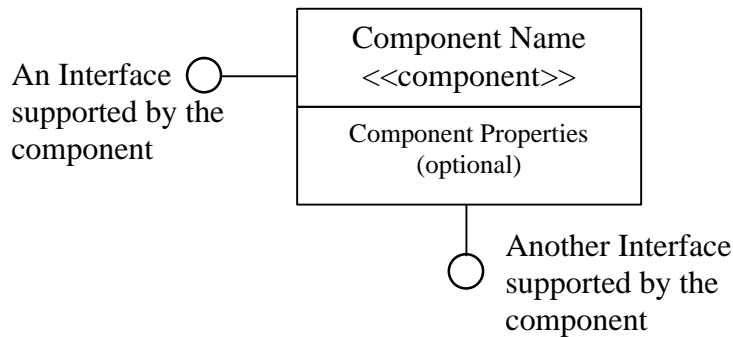Relationship may increase or decrease target

Note that this changes the definition slightly from the book's definition of what a solid arrow and dashed arrow mean. In the book, a solid arrow means that an increase in the originating gauge implies an increase in the target gauge. A dashed arrow means that an increase in the originating gauge implies a decrease in the target gauge. Nothing more is specified. Now, a solid arrow means that the originating and target gauges change in the same direction (up or down) while a dashed arrow means that the originating and target gauges change in the opposite direction. This redefinition does not actually change the diagrams in the book (at least, I don't think so), but allows for more clarity when needed.

Loops in Gauge Diagrams highlight a game's reward systems and balancing mechanisms. Reinforcing Loops have an even number of inverse relationships. Balancing Loops have an odd number of inverse relationships. This isn't an actual change to the technique. It is only an illustration of the concept.

Reinforcing Loops

Balancing Loops
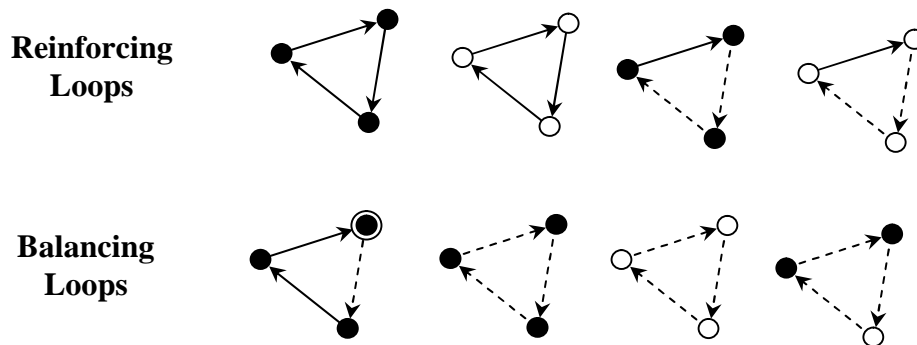
To illustrate game aspects other than gauges and their relationships, other diagramming techniques can be used. UML is particularly useful in this regard.
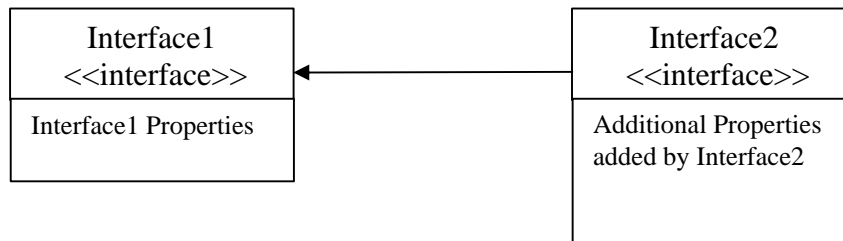
UML Component Diagram Illustrating the Interfaces supported by a component

An Interface supported by the component

Component Name <<component>>
Component Properties (optional)

Another Interface supported by the component

UML Component Diagram illustrating the properties of an interface

Interface Name <<interface>>
Interface Properties
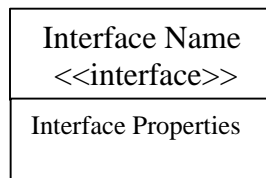
UML Component Diagram Illustrating "is a kind of" relationship

Interface1 <<interface>>
Interface1 Properties

Interface2 <<interface>>
Additional Properties added by Interface2

The arrow illustrates that Interface2 "is a kind of" Interface1. So, it inherits all of Interface1's properties.

UML Component Diagram Example of two interfaces

```
          ┌─────────────────────┐
┌──────────────────┐│     Mage Spell      │
│       Skill      ││    <<interface>>    │
│   <<interface>>  │├─────────────────────┤
├──────────────────┤│ Stress Factor: integer
│ Base: integer    │◄─│ Affected Area: area
│ Description: text ││ Casting Time: time span
│ Prerequisites: text││ Duration: time span
└──────────────────┘│ Range: distance     │
          └─────────────────────┘
```
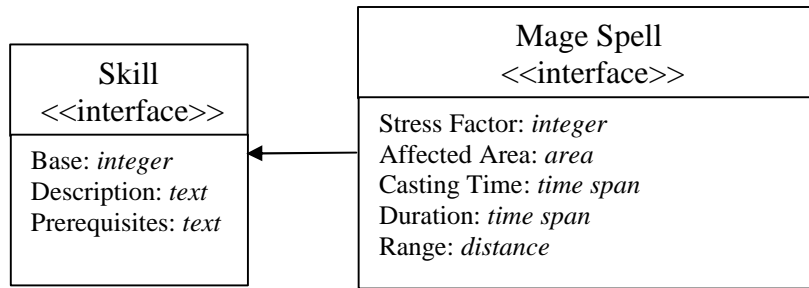
This illustrates that "Skill" and "Mage Spell" are both interfaces. The arrow indicates that "Mage Spell" is a kind of "Skill".

UML Example of a concrete component

The "Mage Spell" icon indicates that "Lightning Bolt" supports the Mage Spell interface.
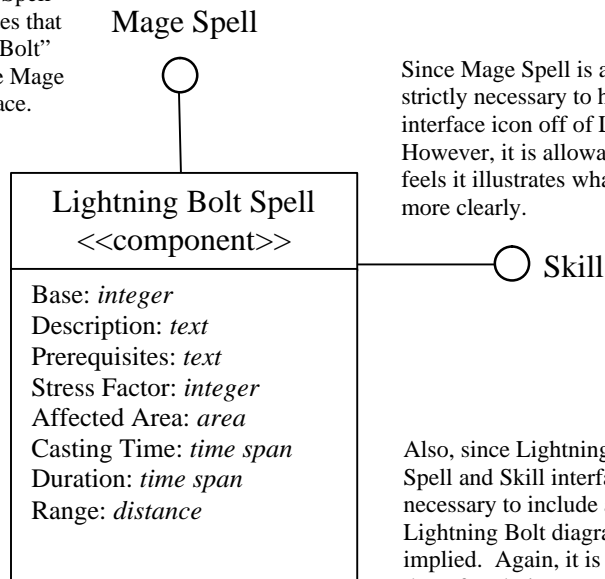
Mage Spell
○

Since Mage Spell is a kind of Skill, it is not strictly necessary to hang this second Skill interface icon off of Lightning Bolt. However, it is allowable if the diagrammer feels it illustrates what he is trying to show more clearly.

```
┌────────────────────────┐
│   Lightning Bolt Spell  │
│      <<component>>      │──────○ Skill
├────────────────────────┤
│ Base: integer          │
│ Description: text       │
│ Prerequisites: text     │
│ Stress Factor: integer  │
│ Affected Area: area     │
│ Casting Time: time span │
│ Duration: time span     │
│ Range: distance         │
└────────────────────────┘
```
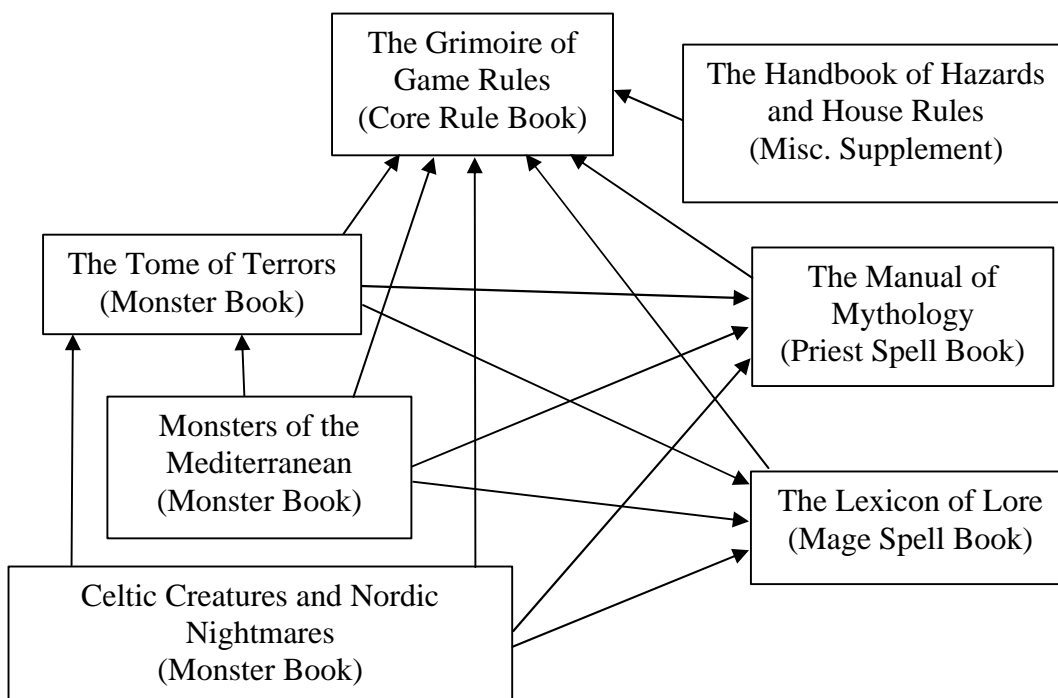
Also, since Lightning Bolt supports the Mage Spell and Skill interfaces, it is not strictly necessary to include all of the properties in the Lightning Bolt diagram. Their inclusion is implied. Again, it is allowable to include them for clarity.

UML Dependency Diagram

| A Module | ← | A Module that depends on another Module |
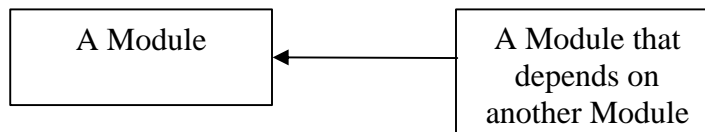
Example UML Dependency Diagram
(Illustrating the dependencies between the various books of Legendary Quest)

- The Grimoire of Game Rules (Core Rule Book)
- The Handbook of Hazards and House Rules (Misc. Supplement)
- The Tome of Terrors (Monster Book)
- The Manual of Mythology (Priest Spell Book)
- Monsters of the Mediterranean (Monster Book)
- The Lexicon of Lore (Mage Spell Book)
- Celtic Creatures and Nordic Nightmares (Monster Book)

Strengths: There are no circular dependencies. It is therefore possible to play the game with *only* the Core Rule Book or a sub-set of the books. (The spell books, although not mandatory for play, are needed by the monster books.) The Core Rule Book has the responsibility of defining all of the interfaces used throughout the game. The supplements, then, merely implement concrete examples of these interfaces.

Weaknesses: Moving a small section out of The Tome of Terrors into The Grimoire of Game Rules would eliminate the dependencies on The Tome of Terrors entirely. In the next edition, I will do this. (This small section is, in essence, the definition of the "Monster" interface.)

Observation: All of the monster books depend on the two spells books as well as the Core Rule Book. This is because a decision was made to write up all specialized monster abilities as spells. This was a conscious decision I made to increase re-use of rules at the expense of adding dependencies between books. (This follows the "Modularity" design pattern.)