Requirements

Powerups are visually shown and can drop from the bubbles. The menu is a start screen where options can be adjusted too.

Must

- Powerups are drawn to the screen with an icon to display that a powerup is in use.
- A powerup has a visual in-game representation that can be picked up by the player.
- On random occasions (once in five splits), the in-game representation of a random powerup drops from a bubble when a bubble is split.

Should

- The shop shows when a powerup is bought with an icon.
- The menu screen is interactive so that options for the game can be set.
- An in-game powerup expires after 30 seconds.
- A power that is bought in the store expires when a level is completed or lost.

Could

- Buttons in the store give visual feedback by adding depth when clicked.
- The player sprite is changed when a powerup is applied.

Would

- The menu is multiple levels deep and screen and audio options can be changed.

CRC

Many of these classes already exist, but we want to look at their responsibilities to see where we apply the changes and lay the new responsibilities.

Classes

- PowerUp
- PowerUpGeneraror
- GameView

PowerUp	
Superclass(es):	
Subclasses: LifePowerUp, SlowPowerUp, TimePowerUp	
Activate powerup	
Draw powerup	GameView

PowerUpGenerator

Superclass(es):	
Subclasses:	
Generate a powerup based on randomness and previous events	Bubble
	PowerUp

GameView	
Superclass(es):	
Subclasses:	
Draw all elements to the screen	Wall
	Bubble
	Player
	PowerUp

Explanations for decisions
We decided to not implement powerups as decorators, because this would limit the effect of powerups to - for example - the player. We'd also like powerups to affect the timer and bubbles.