Modelled `inout`

87

```
class inout
{
public:
    ~inout()
    {
        *cow_vector_arg = cow_vector; // copy back modifications
    }

    // reflect to generate and forward all
    // functions to internal copy

private:
    friend class cow_vector;
    cow_vector* cow_vector_arg; // pointer to original
    cow_vector cow_vector;      // copy, safe to modify

    inout (cow_vector* p)
        : cow_vector_arg (p),
          cow_vector (*p)
    {}
};

inout make_inout()
{
    return inout (this);
}
```

```
void push_42 (cow_vector<int>::inout v)
{
    std::thread t ([v]
                   {
                       // create a copy
                       auto vec2 = *v;
                   });

    v.push_back (42); // no data-race as v has
                      // an internal copy
}
```

```
cow_vector<int> vec;
vec.push_back (40);
vec.push_back (41);

push_42 (&vec);

//... vec contains 42
```

```
inout operator&()
{
    return inout (this);
}
```

Modelled `inout`

```
class inout
{
public:
    ~inout()
    {
        *cow_vector_arg = cow_vector; // copy back modifications
    }

    // reflect to generate and forward all
    // functions to internal copy

private:
    friend class cow_vector;
    cow_vector* cow_vector_arg; // pointer to original
    cow_vector cow_vector;      // copy, safe to modify

    inout (cow_vector* p)
        : cow_vector_arg (p),
          cow_vector (*p)
    {}
};

inout operator&()
{
    return inout (this);
}
```

```
void push_42 (cow_vector<int>::inout v)
{
    std::thread t ([v]
                   {
                       // create a copy
                       auto vec2 = *v;
                   });

    v.push_back (42); // no data-race as v has
                      // an internal copy
}
```

```
cow_vector<int> vec;
vec.push_back (40);
vec.push_back (41);

push_42 (&vec);

//... vec contains 42
```

Mutable Value Semantics