Contract-based Software Development

Rasmus Guldborg Pedersen

January 2015

Overview

1. Code Contracts
2. Class Specification
3. Test or verification
   - Verification
   - Testing

Specification of classes using Code Contracts

What is Code Contract? How can it be used to specify a class?
Does it support test or verification and it what sense?

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

- Static analysis

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

- Static analysis
- Documentation

Code Contracts (.NET tool)

Express preconditions, postconditions and object invariants for:

- Static analysis
- Documentation
- Runtime checking

Interface

```
public interface ISimpleQueue {
    void Enqueue(object item);
    object Dequeue();
    object ElementAt(int index);
    int Count();
}
```

Contract

```
abstract class ISimpleQueueContract {
    public void Enqueue(object item) {
        Contract.Requires(item != null);
        Contract.Ensures(Count ==
            Contract.OldValue(Count()) + 1);
        Contract.Ensures(ElementAt(Count()) == item);
        // ...
    }
    // ...
}
```

Associating Interface with Contract

```
[ContractClass(typeof(ISimpleQueueContract))]
public interface ISimpleQueue { /* ... */ }

[ContractClassFor(typeof(ISimpleQueue))]
abstract class ISimpleQueueContract { /* ... */ }
```

Verification

- $\{Q\}S\{R\}$
- Verify our conceptual model

Testing

- Fail early

Testing

- Fail early
- Test promises

The End

"Testing shows the presence, not the absence of bugs."
— Edsger W. Dijkstra