OO Integration Testing

Chapter 18

- What assumption is made for integration testing?

- What assumption is made for integration testing?
  - **Assume unit level testing is complete**

- What choices are there for unit testing?

- What choices are there for unit testing?
  - **For OO have two choices for unit**

- What choices are there for unit testing?
  - **For OO have two choices for unit**
    - **What are they?**

- What choices are there for unit testing?
  - **For OO have two choices for unit**
    - **Method is a unit**
    - **Class is a unit**

- What does integration testing entail
  - **If method is a unit?**
    - **???**

- What does integration testing entail
  - **If method is a unit?**
    - **Need to integrate within the class**
      - Why?

- What does integration testing entail
  - **If method is a unit?**
    - **Need to integrate within the class**
      - Does occur with classes that have multiple designers / implementers

- What does integration testing entail
  - **If method is a unit?**
    - **Need to integrate within the class**
      - Does occur with classes that have multiple designers / implementers
  - **What else?**

- What does integration testing entail
  - **If method is a unit?**
    - **Need to integrate within the class**
      - Does occur with classes that have multiple designers / implementers
    - **Need to integrate classes**

- What does integration testing entail
  - **If class is a unit?**
    - **???**

- What does integration testing entail
  - **If class is a unit?**
    - **Need to unflatten classes**

- What does integration testing entail
  - **If class is a unit?**
    - **Need to unflatten classes**
  - **What else?**

- What does integration testing entail
  - **If class is a unit?**
    - **Need to unflatten classes**
    - **Need to remove test methods**
  - **What else?**

- What does integration testing entail
  - **If class is a unit?**
    - **Need to unflatten classes**
    - **Need to remove test methods**
    - **Need to integrate classes**

- What considerations are there with integration testing?

- What considerations are there with integration testing?
  - **Static considerations**

- What considerations are there with integration testing?
  - **Static considerations**
    - **What else?**

- What considerations are there with integration testing?
  - **Static considerations**
  - **Dynamic considerations**

- What information do we need for static considerations?

- What information do we need for static considerations?
  - **Class definitions**

- What information do we need for static considerations?
  - **Class definitions**
    - **Where are they?**

- What information do we need for static considerations?
  - **Class definitions**
    - **Program text**

- What information do we need for static considerations?
  - **Class definitions**
    - **Program text**
  - **What else?**

- What information do we need for static considerations?
  - **Class definitions**
    - **Program text**
  - **Static model**

- What information do we need for static considerations?
  - **Class definitions**
    - **Program text**
  - **Static model**
    - **Consists of what?**

- What information do we need for static considerations?
  - **Class definitions**
    - **Program text**
  - **Static model**
    - **Inheritance and uses structure**

- **What tests do we base on static considerations?**
  - **Address polymorphism statically**

- What tests do we base on static considerations?
  - Address polymorphism statically
    - What do we do?

- What tests do we base on static considerations?
  - Address polymorphism statically
    - Select a test for each polymorphic context

- What information do we need for dynamic considerations?
  - **Dynamic view is more challenging**

- What information do we need for dynamic considerations?
  - **Dynamic model**

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Consists of what?**

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Finite state machines – Petri nets**

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Finite state machines – Petri nets**
      - What else?

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Finite state machines – Petri nets**
    - **Class communication – message passing**

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Finite state machines – Petri nets**
    - **Class communication – message passing**
      - What else?

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Finite state machines – Petri nets**
    - **Class communication – message passing**
    - **Use cases – scenarios**
      - What else?

- What information do we need for dynamic considerations?
  - **Dynamic model**
    - **Finite state machines – Petri nets**
    - **Class communication – message passing**
    - **Use cases – scenarios**
      - Statecharts – are not useful

- How do we show class communications?

- How do we show class communications?
  - **Collaboration diagrams**

- How do we show class communications?
  - **Collaboration diagrams**
    - **What else?**

- How do we show class communications?
  - **Collaboration diagrams**
  - **Sequence diagrams**

- What are collaboration diagrams?

- What are collaboration diagrams?
  - **Annotated call graphs – Figure 18.1**

- What are collaboration diagrams?
  - **Annotated call graphs – Figure 18.1**
    - **What types of integration do they support?**

- How do we show class communications?
  - **Collaboration diagrams**
    - **Annotated call graph – Figure 18.1**
  - **Supports**
    - **pair wise integration strategy**
    - **neighbourhood integration strategy**

- What are sequence diagrams?

- What are sequence diagrams?
  - **Finite state machines with time axis – Figure 18.2**

- What are sequence diagrams?
  - **Finite state machines with time axis – Figure 18.2**
    - **What are the states?**

- What are sequence diagrams?
  - **Finite state machines with time axis – Figure 18.2**
    - **States**
      - Classes – regular grain
      - Methods – fine grain

- What are sequence diagrams?
  - **Finite state machines with time axis – Figure 18.2**
    - **States**
      - Classes – regular grain
      - Methods – fine grain
    - **What are the transitions?**

- What are sequence diagrams?
  - **Finite state machines with time axis – Figure 18.2**
    - **States**
      - Classes – regular grain
      - Methods – fine grain
    - **Transitions correspond to sending messages**
  - **What are they analogous to?**

- What are sequence diagrams?
  - **Finite state machines with time axis – Figure 18.2**
    - **States**
      - Classes – regular grain
      - Methods – fine grain
    - **Transitions correspond to sending messages**

  - **Close analogy with MM-paths**

- What types of integration strategies are there?

- What types of integration strategies are there?
  - **Pair-wise**
    - Figure 13.6
  - **Neighbourhood**
    - Figure 13.7

- What is the problem with pair-wise integration?

- What is the problem with pair-wise integration?
  - **Too much extra work with stubs and drivers**

- What is the problem with neighbourhood integration?

- What is the problem with neighbourhood integration?
  - **Some neighbourhoods may include most classes**
  - **Some neighbourhoods may be only two classes**

Figure 18.1

- What is the problem with neighbourhood integration?
  - **Some neighbourhoods may include most classes**
  - **Some neighbourhoods may be only two classes**
  - **What do we do?**

- What is the problem with neighbourhood integration?
  - **Some neighbourhoods may include most classes**
  - **Some neighbourhoods may be only two classes**

  - **What do we do?**
    - **Get a better definition**

- What is a better definition than a neighbourhood?

- What is a better definition than a neighbourhood?
  - **Centers of a graph**
    - **Ultra-center**

- What is a better definition than a neighbourhood?
  - **Centers of a graph**
    - **What properties does an ultra-center have?**

- What is a better definition than a neighbourhood?
  - **Centers of a graph**
    - **Ultra-center**
      - Minimize maximum distance to other nodes
      - Neighbourhood grows from an ultra-center
      - Analogy with ripples from dropping an object into water

- What is a better definition than a neighbourhood?
  - **Centers of a graph**
    - **Ultra-center**
      - Minimize maximum distance to other nodes
      - Neighbourhood grows from an ultra-center
      - Analogy with ripples from dropping an object into water

  - **What are the advantages/disadvantages?**

- What is a better definition than a neighbourhood?
  - **Centers of a graph**
    - **Ultra-center**
      - Minimize maximum distance to other nodes
      - Neighbourhood grows from an ultra-center
      - Analogy with ripples from dropping an object into water
  - **What are the advantages/disadvantages?**
    - **Less stubs**
    - **Less diagnostic precision**

- What is an MM-path – a method to message path – in OO?

- What is an MM-path – a method to message path – in OO?
  - **A sequence of method executions linked by messages**

- What is an MM-path – a method to message path – in OO?
  - **A sequence of method executions linked by messages**
    - **How is an execution path constructed?**

- What is an MM-path – a method to message path – in OO?
  - **A sequence of method executions linked by messages**
    - **Start at any class by sending a message**
    - **End at message quiescence**
    - **End at return from original message**

- What is an MM-path – a method to message path – in OO?
  - **A sequence of method executions linked by messages**
    - **Start at any class by sending a message**
    - **End at message quiescence**
      - What is this?
    - **End at return from original message**

- What is an MM-path – a method to message path – in OO?
  - **A sequence of method executions linked by messages**
    - **Start at any class by sending a message**
    - **End at message quiescence**
      - At class that does not send any messages
    - **End at return from original message**

See Figures 18.3, 18.4, 18.5

- What is the highest integration level?

- What is the highest integration level?
  - **Classes that implement an atomic system function**

Atomic system functions

- What is an atomic system function?

Atomic system functions

- What is an atomic system function?
  - **An MM-path**
    - **Stimulus / response pair of port-level events**

Atomic system functions

- What is an atomic system function?
  - **An MM-path**
    - **Stimulus / response pair of port-level events**
      - What does it begin and end with?

Atomic system functions

- What is an atomic system function?
  - **An MM-path**
    - **Stimulus / response pair of port-level events**
  - **Begins with an input port event**
    - **Event quiescence**
  - **Ends with an output port event**
    - **Event quiescence**

Atomic system functions

- What good are atomic system functions?

Atomic system functions

- What good are atomic system functions?
  - **Addresses event-driven nature of OO programs**
  - **At the boundary of integration and system testing**

OO-calendar analysis

- Why do we use directed graphs?

OO-calendar analysis

- Why do we use directed graphs?
  - **Directed graph makes it possible to be analytical in choosing test cases**

OO-calendar analysis

- How many test cases are there?

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**
    - **What are they?**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**
    - **Start at each of the three statements in routine testIt**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**
    - **Start at each of the three statements in routine testIt**
      - What is the problem?

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**
    - **Start at each of the three statements in routine testIt**
  - **Depends upon choice of test cases, which could miss leap year related cases**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**
    - **Start at each of the three statements in routine testIt**
  - **Depends upon choice of test cases, which could miss leap year related cases**
    - **What do we need to do?**

OO-calendar analysis

- How many test cases are there?
  - **Cyclomatic complexity is 23**
    - **Implies 23 basis paths to test**
  - **Lower bound could be 3 test cases**
    - **Start at each of the three statements in routine testIt**
  - **Depends upon choice of test cases, which could miss leap year related cases**
    - **Need to cover every message**

OO-calendar analysis

- Depends upon choice of test cases, which could miss leap year related cases
  - Need to cover every message
  - What is a good way to do this?

OO-calendar analysis

- **Depends upon choice of test cases, which could miss leap year related cases**
  - **Need to cover every message**
    - The test cases identified in decision table testing (Table 7.16) would give a good integration test suite
    - Look for test cases to cover every message in Figure 18.3

- Are MM-paths sufficient?

Data flow testing

- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**

Data flow testing

- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
    - **Why?**

Data flow testing

- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
  - **Data values add complexity**

Data flow testing

- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
  - **Data values add complexity**
    - **From where does the complexity come?**

Data flow testing

- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
  - **Data values add complexity**
    - **Come from inheritance**
    - **Come from stages of message passing**

Data flow testing

- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
  - **Data values add complexity**
    - **Come from inheritance**
    - **Come from stages of message passing**
  - **What else?**

Data flow testing

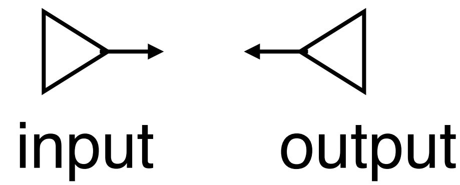
- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
  - **Data values add complexity**
    - **Come from inheritance**
    - **Come from stages of message passing**
  - **Program graphs are basis but are too simple**
    - **What do we need?**

Data flow testing

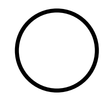
- Are MM-paths sufficient?
  - **Like DD-paths, they are insufficient**
  - **Data values add complexity**
    - **Come from inheritance**
    - **Come from stages of message passing**
  - **Program graphs are basis but are too simple**
    - **Need event and message driven Petri nets**

Event & Message driven Petri nets (EMDPN)

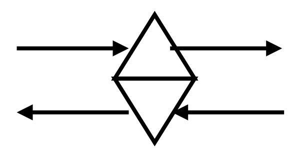
- P – set of port events



- D – set of data places



- M – message send/return places
  - **Output for sender**
  - **Input for receiver**



EMDPN – 2

- T – set of transitions
  - **Represent a method execution path**
- In – set of edges to transitions
  - **$(P \cup D \cup M) \leftrightarrow T$**
    - **It is a relation between places and transitions**
    - **If deterministic then it is a function from places to transitions**
- Out – set of edges from transitions
  - **$T \leftrightarrow (P \cup D \cup M)$**

Message send/receive places

- Capture notion of **interobject** messages
  - **They are an sink of a method execution path in the sending object**
  - **They are an source to a method execution path in the receiving object**
  - **The return is an sink of a method execution path in the receiving object**
  - **The return is an source to a method execution path in the sending object**

See Figure 18.7

DU-paths

- Define / use paths
  - **Focus on connectivity**
  - **Ignore types of nodes**

Inheritance-induced data flow

- Begins with a data place
- Ends with a data place
- Data places alternate with isA transitions
  - **isA transitions are degenerate execution paths**
    - **Implement inheritance**

See Figure 18.8

Message-induced data flow

- Set of transitions
  - **Start with defining transition**
    - **Variable is defined in the module execution path**
  - **End with use transition**
    - **Variable Is used in the module execution path**
- Can be definition clear or not definition clear

See Figure 18.9
&
Section 18.3.3 for an example path

Slices

- Useful if executable
  - **Difficult to do in OO environment**

- Can be used for desk checking for fault location