

FIT3013 Lecture Notes

Introduction to Event-B

John Hurst, 2009

Computer Science and Software Engineering
Monash University
(with acknowledgement to Ken Robinson, UNSW)

20090302 / Lecture 1

Outline

An Introduction to Event-B

Comparison with other Engineerings

Advantages of the B Method

Practical Uses

Summary

An Introduction to Event-B

- In this course we will be introducing you to the formal method, whose full name is the B Method, but which is usually known simply as B.
- The B notation has recently been revised to a form called Event-B, and it is this method and its associated tool Rodin, that we will be studying.
- B (is one of the few formal software development methods that covers the complete software lifecycle, from requirements (specification), through design (refinement) to implementation, code generation, and maintenance.

An Introduction to Event-B

- In this course we will be introducing you to the formal method, whose full name is the B Method, but which is usually known simply as B.
- The B notation has recently been revised to a form called Event-B, and it is this method and its associated tool Rodin, that we will be studying.
- B (is one of the few formal software development methods that covers the complete software lifecycle, from requirements (specification), through design (refinement) to implementation, code generation, and maintenance.

An Introduction to Event-B

- In this course we will be introducing you to the formal method, whose full name is the B Method, but which is usually known simply as B.
- The B notation has recently been revised to a form called Event-B, and it is this method and its associated tool Rodin, that we will be studying.
- **B** (is one of the few formal software development methods that covers the complete software lifecycle, from requirements (specification), through design (refinement) to implementation, code generation, and maintenance.

An Introduction to Event-B

- The Event-B tool Rodin has yet to be modified to include implementation – however, this will not affect the material studied in this course.
- In this course we will be concerned only with the use of Event-B for specification.
- To assist developments in Event-B, we will be use the Rodin Toolkit, a configuration management tool for Event-B.

An Introduction to Event-B

- The Event-B tool Rodin has yet to be modified to include implementation – however, this will not affect the material studied in this course.
- In this course we will be concerned only with the use of Event-B for specification.
- To assist developments in Event-B, we will be use the Rodin Toolkit, a configuration management tool for Event-B.

An Introduction to Event-B

- The Event-B tool Rodin has yet to be modified to include implementation – however, this will not affect the material studied in this course.
- In this course we will be concerned only with the use of Event-B for specification.
- To assist developments in Event-B, we will be use the Rodin Toolkit, a configuration management tool for Event-B.

What do we mean by Formal Methods?

- In our use of the term **Formal Methods** we mean the application of mathematics –**set theory and logic**– to specify, design and implement software –or more generally systems– in such a way that the resulting code has been **proved** to be **consistent** with the original specification.
- In Event-B, a specification is a mathematical model of the required behaviour of a system. Specifications are generally abstract.
- We then transform the specification through a sequence of formally defined **refinement** steps towards a **concrete** implementation.
- During the process we have a number of **proof obligations** that must be **discharged**.

What do we mean by Formal Methods?

- In our use of the term **Formal Methods** we mean the application of mathematics –**set theory and logic**– to specify, design and implement software –or more generally systems– in such a way that the resulting code has been **proved** to be **consistent** with the original specification.
- In Event-B, a specification is a mathematical model of the required behaviour of a system. Specifications are generally abstract.
- We then transform the specification through a sequence of formally defined **refinement** steps towards a **concrete** implementation.
- During the process we have a number of **proof obligations** that must be **discharged**.

What do we mean by Formal Methods?

- In our use of the term **Formal Methods** we mean the application of mathematics –**set theory and logic**– to specify, design and implement software –or more generally systems– in such a way that the resulting code has been **proved** to be **consistent** with the original specification.
- In Event-B, a specification is a mathematical model of the required behaviour of a system. Specifications are generally abstract.
- We then transform the specification through a sequence of formally defined **refinement** steps towards a **concrete** implementation.
- During the process we have a number of **proof obligations** that must be **discharged**.

What do we mean by Formal Methods?

- In our use of the term **Formal Methods** we mean the application of mathematics –**set theory and logic**– to specify, design and implement software –or more generally systems– in such a way that the resulting code has been **proved** to be **consistent** with the original specification.
- In Event-B, a specification is a mathematical model of the required behaviour of a system. Specifications are generally abstract.
- We then transform the specification through a sequence of formally defined **refinement** steps towards a **concrete** implementation.
- During the process we have a number of **proof obligations** that must be **discharged**.

Connection with “conventional” methods

- **Conventional software development methods usually express the requirements informally:**
 - either structured or unstructured English,
 - or using some structured notation: dataflow, entity relationship diagrams,
 - or the unified modelling language (UML).
- Specifications are frequently expressed directly in programming code.
- Design then consists of “fleshing out” the code to produce an implementation.

Connection with “conventional” methods

- Conventional software development methods usually express the requirements informally:
 - **either structured or unstructured English,**
 - or using some structured notation: dataflow, entity relationship diagrams,
 - or the unified modelling language (UML).
- Specifications are frequently expressed directly in programming code.
- Design then consists of “fleshing out” the code to produce an implementation.

Connection with “conventional” methods

- Conventional software development methods usually express the requirements informally:
 - either structured or unstructured English,
 - or using some structured notation: dataflow, entity relationship diagrams,
 - or the unified modelling language (UML).
- Specifications are frequently expressed directly in programming code.
- Design then consists of “fleshing out” the code to produce an implementation.

Connection with “conventional” methods

- Conventional software development methods usually express the requirements informally:
 - either structured or unstructured English,
 - or using some structured notation: dataflow, entity relationship diagrams,
 - or the unified modelling language (UML).
- Specifications are frequently expressed directly in programming code.
- Design then consists of “fleshing out” the code to produce an implementation.

Connection with “conventional” methods

- Conventional software development methods usually express the requirements informally:
 - either structured or unstructured English,
 - or using some structured notation: dataflow, entity relationship diagrams,
 - or the unified modelling language (UML).
- Specifications are frequently expressed directly in programming code.
- Design then consists of “fleshing out” the code to produce an implementation.

Connection with “conventional” methods

- Conventional software development methods usually express the requirements informally:
 - either structured or unstructured English,
 - or using some structured notation: dataflow, entity relationship diagrams,
 - or the unified modelling language (UML).
- Specifications are frequently expressed directly in programming code.
- Design then consists of “fleshing out” the code to produce an implementation.

Connection with “conventional” methods

(cont. 2)

- In contrast, when using a formal development method, like **Event-B**, the specification is an abstract description of the requirements, expressing **what** behaviour is required, rather than **how** to produce that behaviour.
- It follows the design phase must effect a radical transformation of the specification in order to obtain executable code.

Connection with “conventional” methods

(cont. 2)

- In contrast, when using a formal development method, like **Event-B**, the specification is an abstract description of the requirements, expressing **what** behaviour is required, rather than **how** to produce that behaviour.
- It follows the design phase must effect a radical transformation of the specification in order to obtain executable code.

Testing within traditional engineering disciplines

- Consider traditional engineering disciplines, such as electrical engineering, or civil engineering
- Designs are based on a mathematical theory of materials, components, used in the implementation of bridges, electronic circuits, etc..
- Testing consists of physical testing of the implementation, or a model of the implementation.
- Successful strategy because domains are described by continuous mathematics: if a model conforms for some specific test input, it will also conform for input that is “less than” that input. Hence we need only test for extreme values.
- This strategy would not work for discrete valued domains.

Testing within traditional engineering disciplines

- Consider traditional engineering disciplines, such as electrical engineering, or civil engineering
- Designs are based on a mathematical theory of materials, components, used in the implementation of bridges, electronic circuits, etc..
- Testing consists of physical testing of the implementation, or a model of the implementation.
- Successful strategy because domains are described by continuous mathematics: if a model conforms for some specific test input, it will also conform for input that is “less than” that input. Hence we need only test for extreme values.
- This strategy would not work for discrete valued domains.

Testing within traditional engineering disciplines

- Consider traditional engineering disciplines, such as electrical engineering, or civil engineering
- Designs are based on a mathematical theory of materials, components, used in the implementation of bridges, electronic circuits, etc..
- Testing consists of physical testing of the implementation, or a model of the implementation.
- Successful strategy because domains are described by continuous mathematics: if a model conforms for some specific test input, it will also conform for input that is “less than” that input. Hence we need only test for extreme values.
- This strategy would not work for discrete valued domains.

Testing within traditional engineering disciplines

- Consider traditional engineering disciplines, such as electrical engineering, or civil engineering
- Designs are based on a mathematical theory of materials, components, used in the implementation of bridges, electronic circuits, etc..
- Testing consists of physical testing of the implementation, or a model of the implementation.
- Successful strategy because domains are described by continuous mathematics: if a model conforms for some specific test input, it will also conform for input that is “less than” that input. Hence we need only test for extreme values.
- This strategy would not work for discrete valued domains.

Testing within traditional engineering disciplines

- Consider traditional engineering disciplines, such as electrical engineering, or civil engineering
- Designs are based on a mathematical theory of materials, components, used in the implementation of bridges, electronic circuits, etc..
- Testing consists of physical testing of the implementation, or a model of the implementation.
- Successful strategy because domains are described by continuous mathematics: if a model conforms for some specific test input, it will also conform for input that is “less than” that input. Hence we need only test for extreme values.
- **This strategy would not work for discrete valued domains.**

Contrast with testing of software

- Software executes over discrete domains, and testing usually consists of probing points within that space. Thus testing can only confirm conformance of behaviour at specific points. Testing is therefore incapable, in general, of demonstrating conformance over the complete application domain.
- Thus testing may confirm the presence of bugs, but not their absence.

Contrast with testing of software

- Software executes over discrete domains, and testing usually consists of probing points within that space. Thus testing can only confirm conformance of behaviour at specific points. Testing is therefore incapable, in general, of demonstrating conformance over the complete application domain.
- Thus testing may confirm the presence of bugs, but not their absence.

Advantages of formal development and proof

- Using a formal development method, build a model using constructs described by precise mathematical theories. These models capture the behaviour in a complete application domain.
- As we develop our specifications into implementations, the formal method produces proof obligations that describe the complete set of tests that confirm that the behaviours of the specification and the design are consistent –more correctly, not inconsistent.
- Discharging the proof obligations is thus the counterpart of testing in other engineering disciplines.
- A proof validates behaviour in a complete domain, not simply at a single point.

Advantages of formal development and proof

- Using a formal development method, build a model using constructs described by precise mathematical theories. These models capture the behaviour in a complete application domain.
- As we develop our specifications into implementations, the formal method produces proof obligations that describe the complete set of tests that confirm that the behaviours of the specification and the design are consistent –more correctly, not inconsistent.
- Discharging the proof obligations is thus the counterpart of testing in other engineering disciplines.
- A proof validates behaviour in a complete domain, not simply at a single point.

Advantages of formal development and proof

- Using a formal development method, build a model using constructs described by precise mathematical theories. These models capture the behaviour in a complete application domain.
- As we develop our specifications into implementations, the formal method produces proof obligations that describe the complete set of tests that confirm that the behaviours of the specification and the design are consistent –more correctly, not inconsistent.
- Discharging the proof obligations is thus the counterpart of testing in other engineering disciplines.
- A proof validates behaviour in a complete domain, not simply at a single point.

Advantages of formal development and proof

- Using a formal development method, build a model using constructs described by precise mathematical theories. These models capture the behaviour in a complete application domain.
- As we develop our specifications into implementations, the formal method produces proof obligations that describe the complete set of tests that confirm that the behaviours of the specification and the design are consistent –more correctly, not inconsistent.
- Discharging the proof obligations is thus the counterpart of testing in other engineering disciplines.
- **A proof validates behaviour in a complete domain, not simply at a single point.**

What has been done with formal methods?

- Formal methods have been used in various mission critical applications, for example train control systems and smart cards.
- In some countries the use of formal methods is mandatory for certain critical systems.
- The most famous and significant use of B is for the control system of the Météor line (Paris Metro Line 14), a new driverless train line of the Paris metro opened in October 1998.

What has been done with formal methods?

- Formal methods have been used in various mission critical applications, for example train control systems and smart cards.
- In some countries the use of formal methods is mandatory for certain critical systems.
- The most famous and significant use of B is for the control system of the Météor line (Paris Metro Line 14), a new driverless train line of the Paris metro opened in October 1998.

What has been done with formal methods?

- Formal methods have been used in various mission critical applications, for example train control systems and smart cards.
- In some countries the use of formal methods is mandatory for certain critical systems.
- The most famous and significant use of B is for the control system of the Météor line (Paris Metro Line 14), a new driverless train line of the Paris metro opened in October 1998.

More Industrial Pages

- See

<http://www.univ-valenciennes.fr/ROI/WP/download/2003/03-2003.pdf>

http://en.wikipedia.org/wiki/Paris_Metro_Line_14



What has been done with formal methods?

(cont. 2)

- The distributed control system handled the critical parts of the central control room, the wayside equipment along the track and on the platform, the onboard train control.
- The use of B was mandated by the Paris transit authority, RATP. The Météor system was developed by Matra Transport, now owned by Siemens.

What has been done with formal methods?

(cont. 2)

- The distributed control system handled the critical parts of the central control room, the wayside equipment along the track and on the platform, the onboard train control.
- The use of B was mandated by the Paris transit authority, RATP. The Météor system was developed by Matra Transport, now owned by Siemens.

More Industrial Pages

- The following pages on the Siemens site mention the B-Method in connection with Safety and Innovation

`http://www.siemens-ts.com/pagesUS/Engagements/
{Securite.htm, Innovation.htm}`

- The following page talks about the announcement of the award of contract to install the Météor system on the New York subway.

`http://www.siemens-ts.com/pagesUS/realisations/NewYork.htm`

- Or, see the pages at

`http://www.gemplus.com/smart/r_d/trends/system_model/b_method/
GEMPLUS is a company that develops smart cards.`

More Industrial Pages

- The following pages on the Siemens site mention the B-Method in connection with Safety and Innovation
`http://www.siemens-ts.com/pagesUS/Engagements/Securite.htm`, `Innovation.htm`
- The following page talks about the announcement of the award of contract to install the Météor system on the New York subway.
`http://www.siemens-ts.com/pagesUS/realisations/NewYork.htm`
- Or, see the pages at
`http://www.gemplus.com/smart/r_d/trends/system_model/b_method/`
GEMPLUS is a company that develops smart cards.

More Industrial Pages

- The following pages on the Siemens site mention the B-Method in connection with Safety and Innovation
`http://www.siemens-ts.com/pagesUS/Engagements/{Securite.htm, Innovation.htm}`
- The following page talks about the announcement of the award of contract to install the Météor system on the New York subway.
`http://www.siemens-ts.com/pagesUS/realisations/NewYork.htm`
- Or, see the pages at
`http://www.gemplus.com/smart/r_d/trends/system_model/b_method/`
GEMPLUS is a company that develops smart cards.

Formality is inexorable

- The increase in the use of formality in software development has been continuous, from
 - formal grammars to specify programming language syntax, to
 - the semi-formal application of translator generators in compiler implementation.
- High-level programming languages themselves are an instance of increased formality, over machine level (assembler) programming in this case.

Formality is inexorable

- The increase in the use of formality in software development has been continuous, from
 - formal grammars to specify programming language syntax, to
 - the semi-formal application of translator generators in compiler implementation.
- High-level programming languages themselves are an instance of increased formality, over machine level (assembler) programming in this case.

Formality is inexorable

- The increase in the use of formality in software development has been continuous, from
 - formal grammars to specify programming language syntax,to
 - the semi-formal application of translator generators in compiler implementation.
- High-level programming languages themselves are an instance of increased formality, over machine level (assembler) programming in this case.

Formality is inexorable

- The increase in the use of formality in software development has been continuous, from
 - formal grammars to specify programming language syntax, to
 - the semi-formal application of translator generators in compiler implementation.
- High-level programming languages themselves are an instance of increased formality, over machine level (assembler) programming in this case.

Formality is inexorable

(cont. 2)

- Everywhere rigour and formality has been used there has been an increase in the reliability of implementations.
- There is no reason to believe that this “progress” will not continue.

Formality is inexorable

(cont. 2)

- Everywhere rigour and formality has been used there has been an increase in the reliability of implementations.
- There is no reason to believe that this “progress” will not continue.

Summary of Main Points

- The Event-B Method is a **formal method** for software development.
- Here focus on use of Event-B for **specification**.
- Formal methods bring **mathematical rigour** to software development.
- Formal methods have been used to build practical computer systems.

Summary of Main Points

- The Event-B Method is a **formal method** for software development.
- Here focus on use of Event-B for **specification**.
- Formal methods bring **mathematical rigour** to software development.
- Formal methods have been used to build practical computer systems.

Summary of Main Points

- The Event-B Method is a **formal method** for software development.
- Here focus on use of Event-B for **specification**.
- Formal methods bring **mathematical rigour** to software development.
- Formal methods have been used to build practical computer systems.

Summary of Main Points

- The Event-B Method is a **formal method** for software development.
- Here focus on use of Event-B for **specification**.
- Formal methods bring **mathematical rigour** to software development.
- Formal methods have been used to build practical computer systems.