Software Engineering with Formal Methods :

The Development of a

Storm Surge Barrier Control System

Seven Myths of Formal Methods Revisited

Jan Tretmans
University of Twente

Klaas Wijbrans, Michel Chaudron
CMG Public Sector B.V.

With support from:
Franc Buve, Eric Burgers, Wouter Geurts,
Rijn Buve, Sjaak de Graaf, Hedde van de Lugt,
Peter Bosman, Peter van de Heuvel,
Robin Rijkers, Pim Kars, Ed Brinksma,
Wil Janssen, Theo Ruys, Job Zwiers

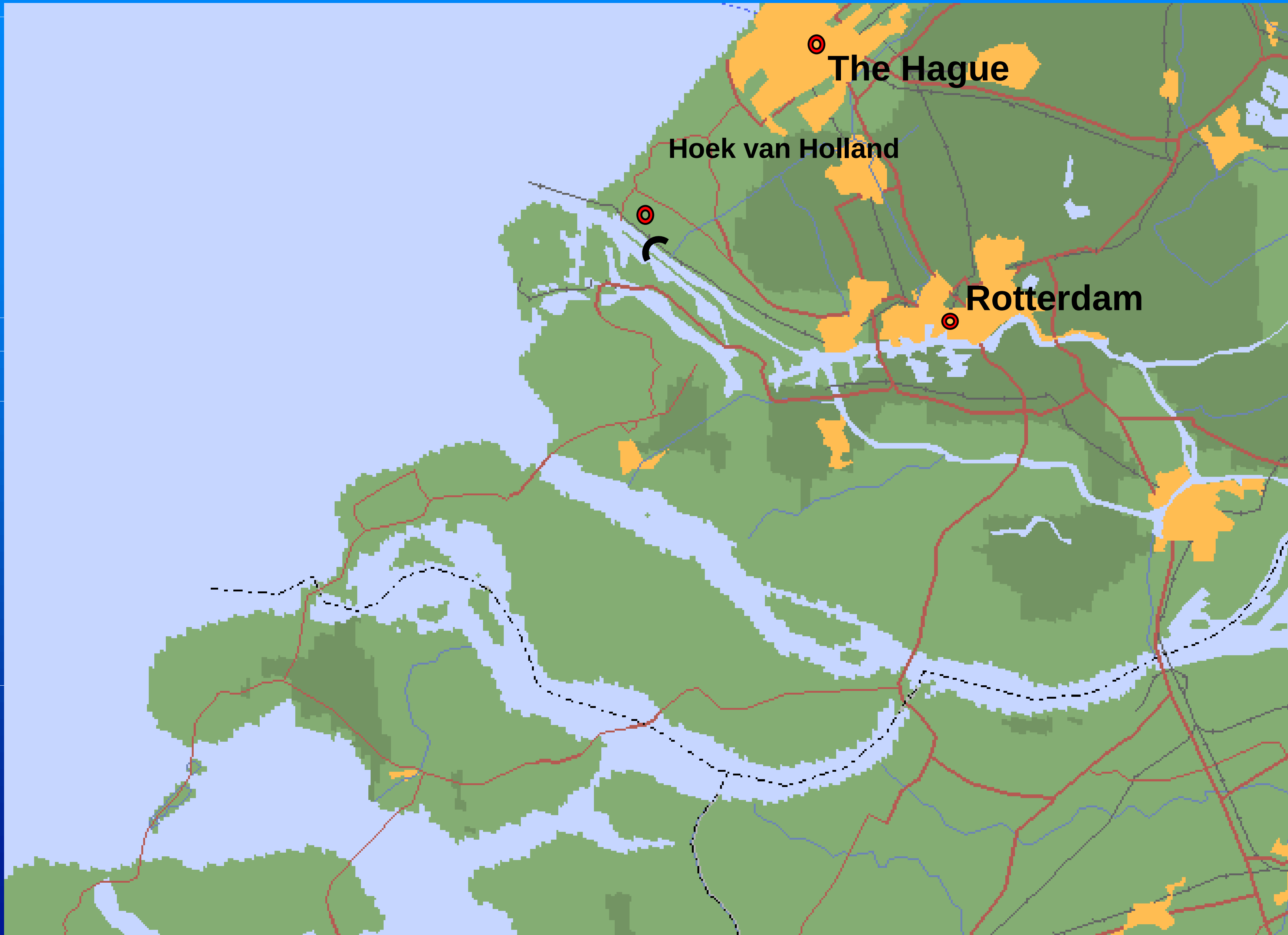
Overview

- The storm surge barrier and BOS
- Approach to Software Development
- Experiences with the use of formal methods:

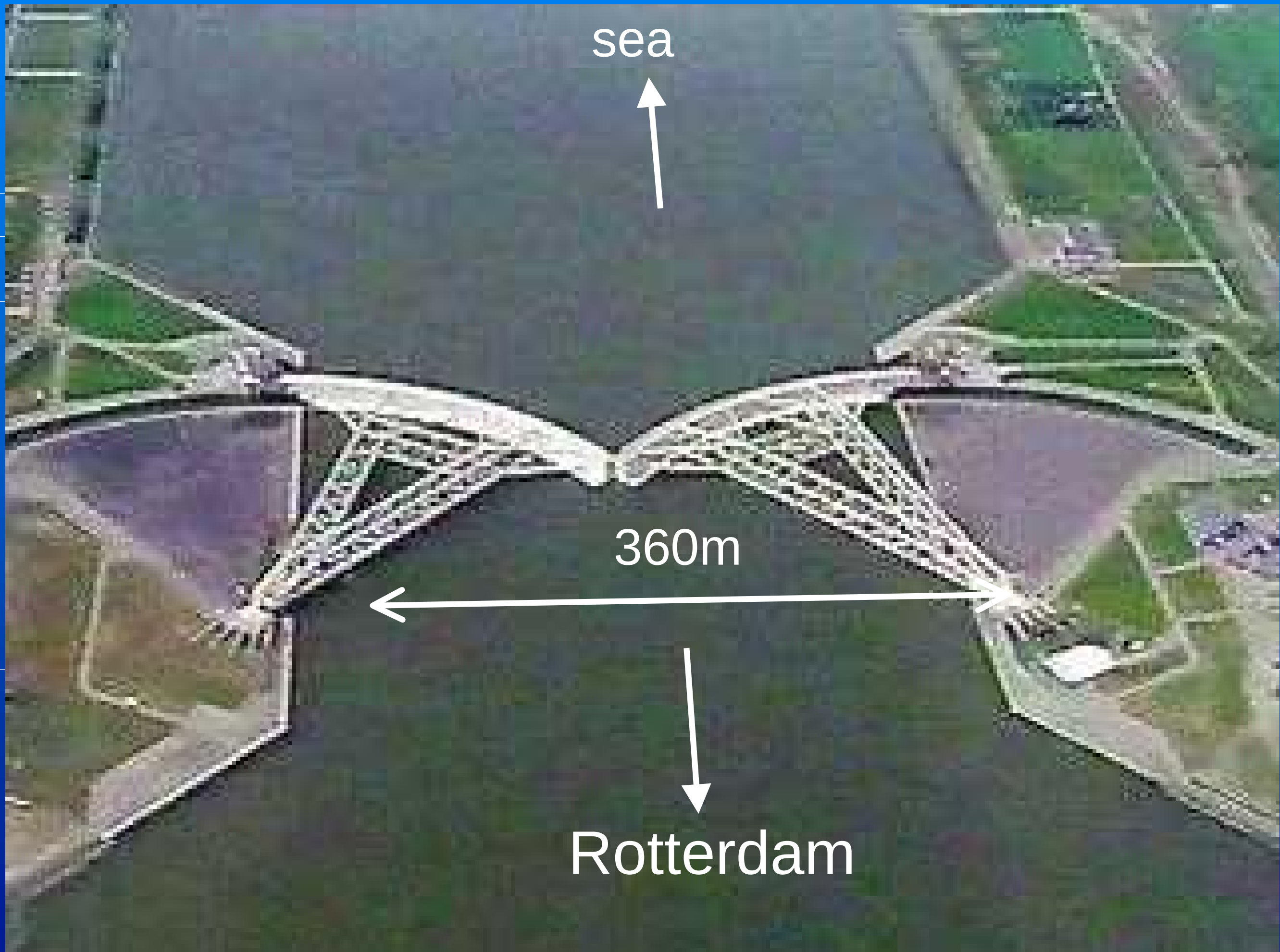
  Seven Myths of Formal Methods revisited

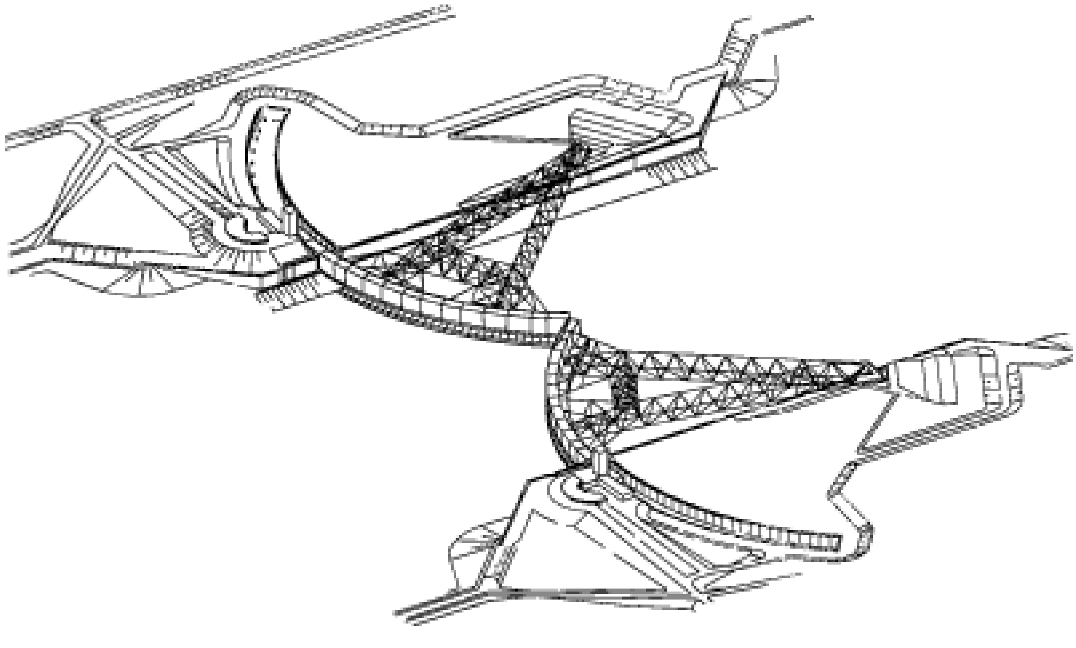
  (*Seven Myths of Formal Methods*,
  Anthony Hall, IEEE Software 6(9), 1990)

Hamburg
Berlin
Hannover
POLAN
NETHERLAND
University of Twente
The Netherlands
Düsseldorf
GERMANY
Brussels
Cologne
BELGIUM
Elbe
Prague
CZECH REPUBLI
Paris
Vienna
Munich
Danube
Loire
AUSTRIA
ALPS

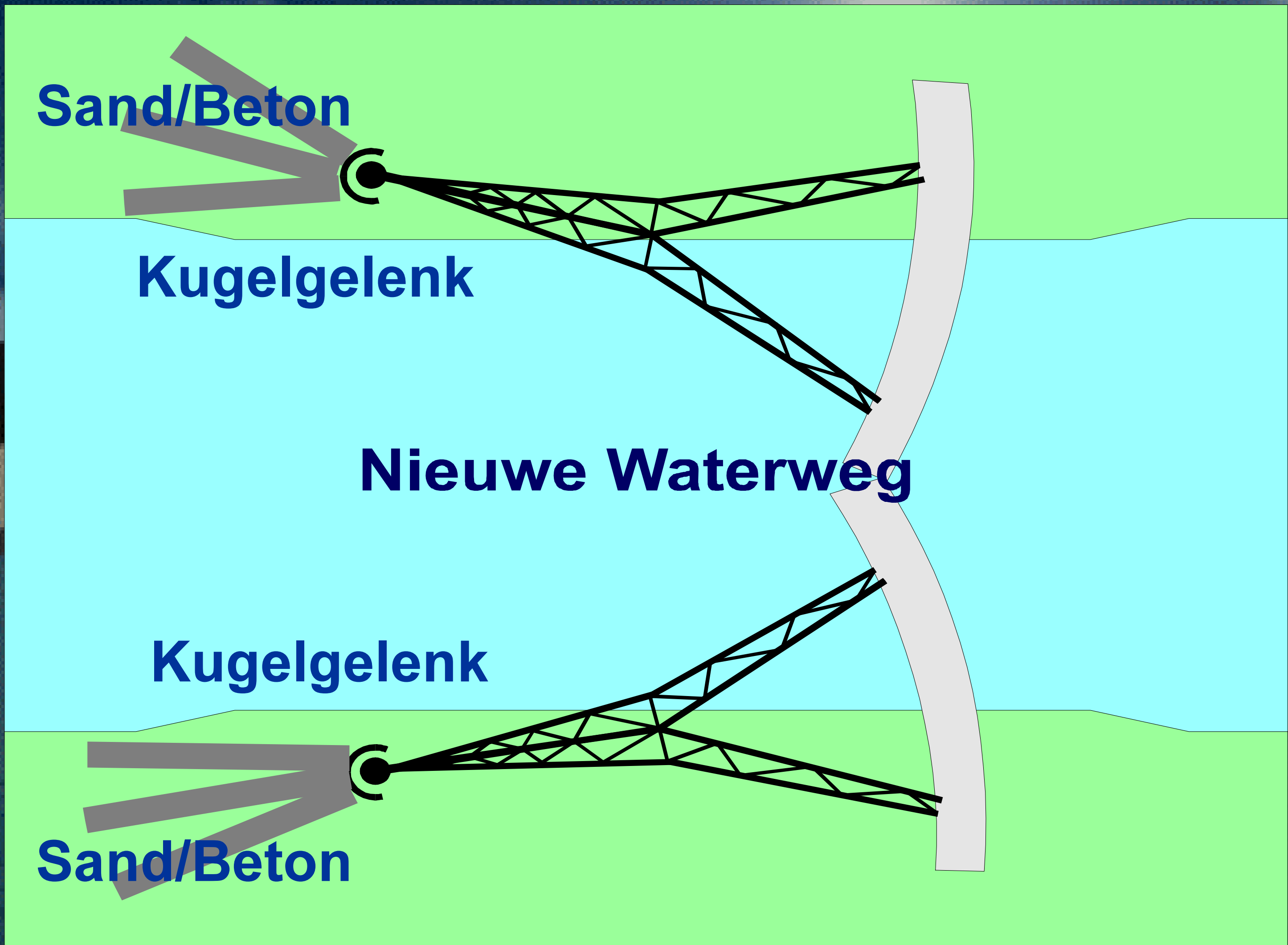

The Hague
Hoek van Holland
Rotterdam

Storm surge barrier



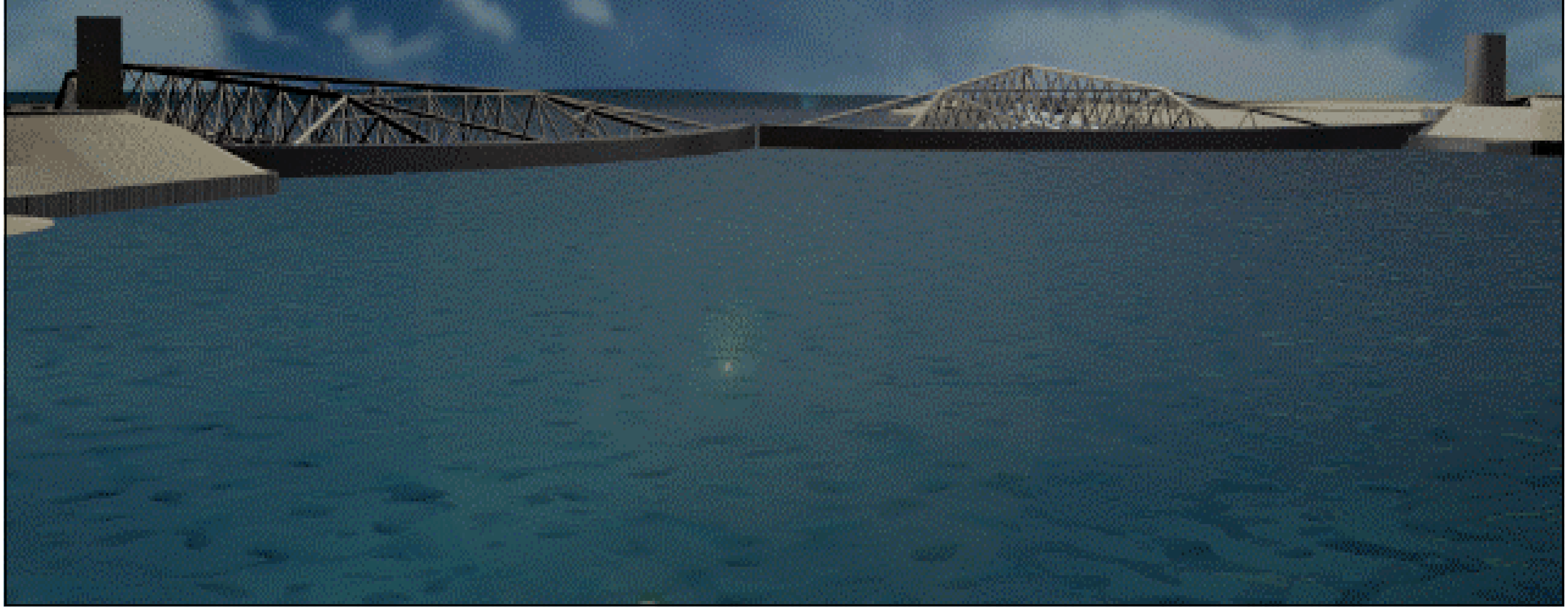


The storm surge barrier



As reliable as a dyke

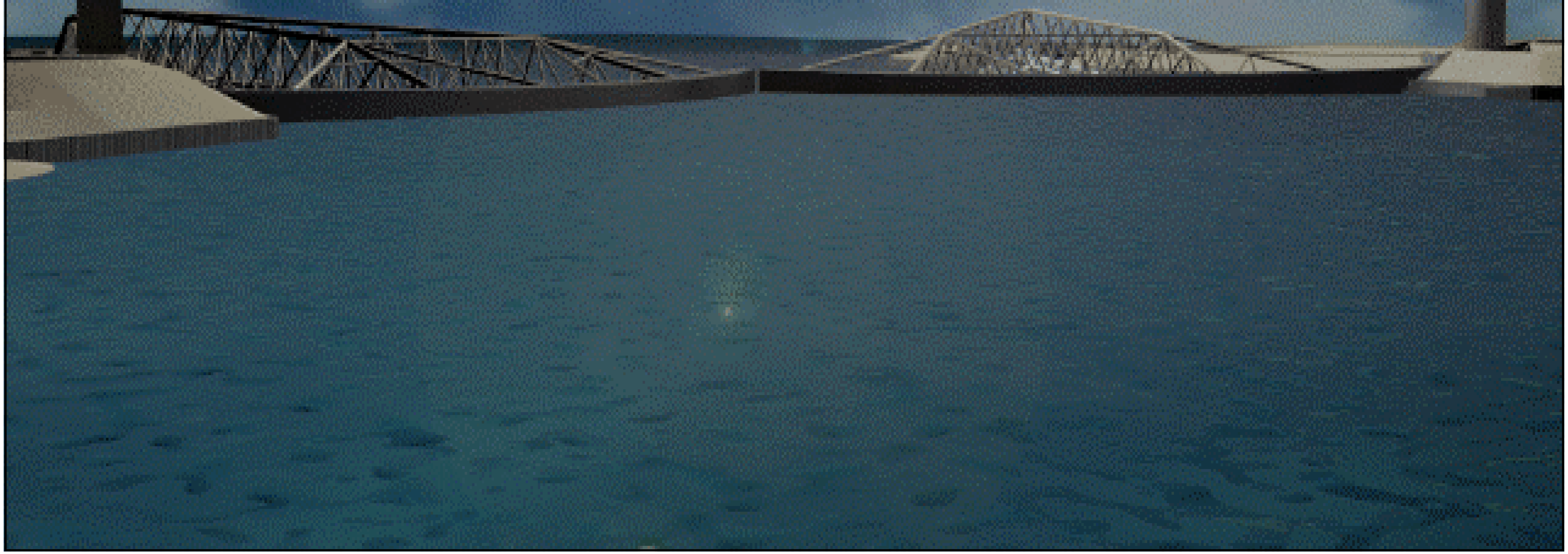
- Risk of failure dike: 1 in 10.000 years
- Extreme high water: 1 in 10 years
- Risk of failure barrier: 1 in 1.000 closures



Why a BOS?

- Risk of failure in decision: 1 in 100.000
- Failure probability human: 1 in 1.000 -10.000
- Decision process must be done automatically

=> BOS



What does BOS do?

- Decide when to close the barrier
- Decide when to open the barrier
- Control the barrier in the Nieuwe Waterweg
- Control the barrier in the Hartelkanaal
- Decide when maintenance is allowed
- Decide if a test closure is allowed and perform it

How does it work?

- Acquire data
- Predict water levels
- Decide and control
- Archive

How was it built?

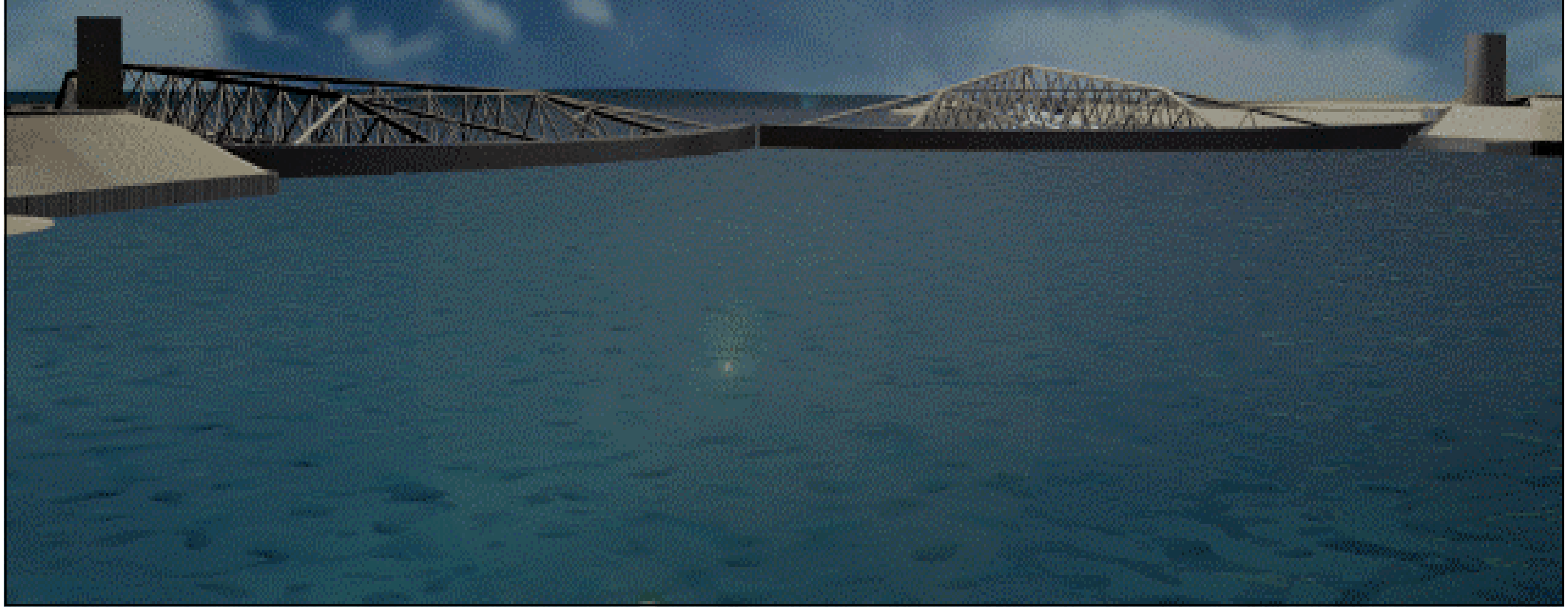
- Functional requirements drawn by RWS
- Required reliability 1 in 100.000
- Technical design and software by CMG (fixed-price)
- 25 person-years, 3 years from start to end
- 450.000 lines of code

High reliability

- Hardware
  - Fault-tolerant Stratus Continuum computer
  - Multiple communication lines and a private satellite channel
- Software
  - Reliability 1 in 100.000 cannot be shown by testing: it would take 2000 year!
  - Certification according to ISO 9001
  - Standard for safety-critical systems: IEC 1508

Result

- Delivered october 1997 within budget and on-time
- Since then it has rightly been in alert twice
- On October 3, 1998 the first test closure



Standard IEC 61508

- Recommendation for implementing safety critical software

- No measure of reliability or safety for software

- Based on "Best Practices"

- Recommends and forbids particular development techniques based on *Safety Integrity Level* SIL:

P_{fail}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
SIL	1	2	3	4
				$\Uparrow$ BOS

- *Highly Recommended* for SIL 4:
    - inspections
    - independent testing
    - .....
    - *formal methods*

BOS Approach

- Risk based / failure analysis
- Well-defined project life-cycle, strict configuration management
- Use of best, feasible practices
- No guarantee of correctness, but increase of confidence by combination of validation techniques: *Validation & Verification Plan*
- V & V plan:
  - reviews and inspections
  - coding standards and static checking
  - coding assertions
  - developer testing (white box)
  - independent module/integration/system testing with coverage measurement
  - formal specifications
  - model checking
  - simulation

Formal Methods in BOS

Starting Points

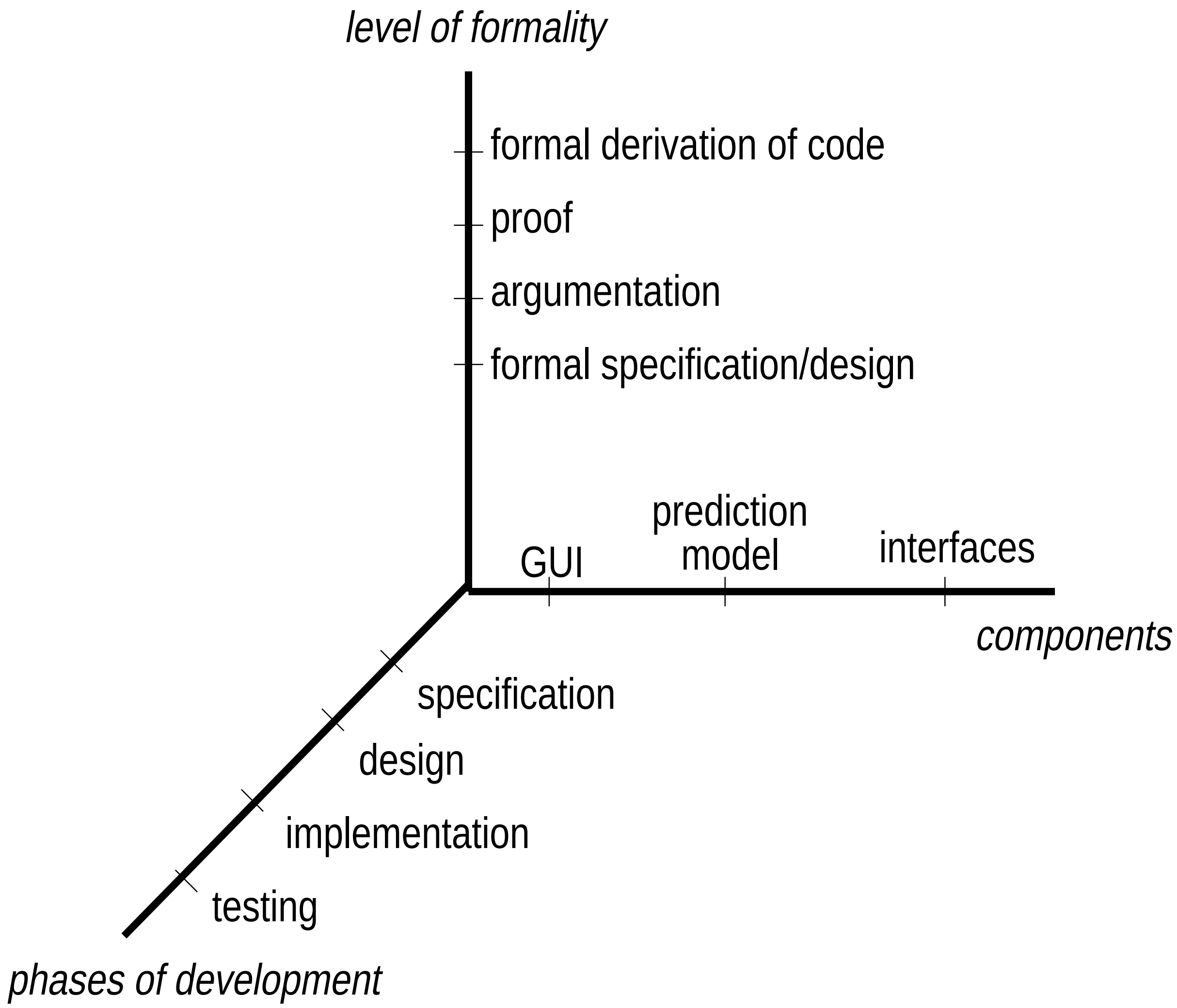
- Goal:
  To increase the reliability and correctness of the BOS software
- Cooperation CMG – University of Twente
- Integration in development trajectory
  not a parallel, academic exercise
- Fixed time – fixed price project
- Non-formal (Dutch + Hatley & Pirbhai) specification given

Formal Methods in BOS

How to Start ?

- where to apply ?
- which formal techniques ?
- in which phases of development ?
- how to learn FM ?
- how to manage FM ?
- how to combine with other (non-formal) SE techniques ?
- expected costs and benefits ?

Where to apply FM



Seven Myths of Formal Methods

Are the *seven myths of formal methods* really myths for the BOS project ?

(From: *Seven Myths of Formal Methods*, Anthony Hall, IEEE Software 6(9), 1990)

1. *FM guarantee correctness*
2. *FM are about program proving*
3. *FM are only for safety-critical systems*
4. *FM require highly trained mathematicians*
5. *FM increase development costs*
6. *FM are unacceptable to users*
7. *FM are not used on real software*

Myth 1

FM guarantee correctness

BOS: no guarantee of correctness

increased confidence in correctness
through increased precision and
early detection of defects

No guarantee of correctness:

- No single method can guarantee correctness

- FM not applied with all their rigour:
  - System too large for manual proof
  - No usable tools for automatic proof
  - Don't be too formal during first time use of FM

- Not all aspects of behaviour in single formalism

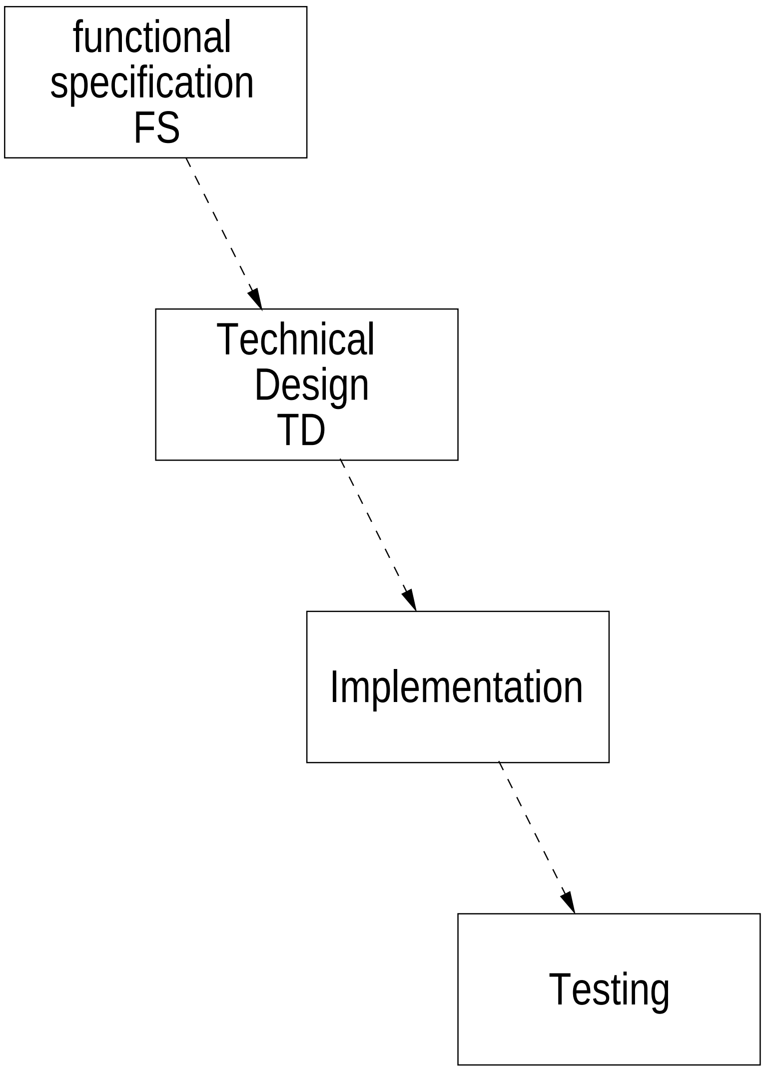
- Not all system components formally specified

- Difference formal model $\leftrightarrow$ reality

- Specification not formal

- Implementation and testing based on formal specifications but no formal derivation or proof

Development Trajectory



Myth 2

FM are about program proving

BOS: no program proved, but

- ∘ formal description of design
- ∘ model checking of some protocols

Result:

- process of formally specifying
  leads to early detection of errors
- precise, unambiguous, complete
- understanding, argumentation and simulation
- precise basis for implementation and testing

Early phases most critical;
implementation / testing no big problem

Myth 3

FM are only for safety-critical systems

BOS: safety-critical and FM used

Other systems:

- almost all systems critical –
  safety, economic, company image

- BOS experience:
  better quality with (almost) same costs

- lot of costs in learning

Myth 4

FM require highly trained mathematicians

BOS: software engineers can learn FM

- FM based on relatively simple mathematics
- No need to learn all mathematical background for using FM
- No complete proofs
  Model checking by few persons
- No mathematicians, please:
  - If they have a model they can calculate
    Problem: to get the valid model
  - they aim at "elegant" solutions
    not at practical ones
- Not mathematics of FM is difficult
  but learning effective usage in SE

Learning FM

- learning formal languages not difficult
- learning formal method difficult
- difficulties:
  - how to make models / level of abstraction
  - what to formalize and what not
  - how to manage FM
  - how to use the tools effectively
  - attitude and mentality change
  - developing specification styles / standards
  - developing coding techniques
  - developing reviewing techniques for FM
  - embedding in SE process
    integration with existing methods

Learning FM

- courses only to learn the formal language

- university knowledge restricted to formal language knowledge

- learning *usage* on the job

- potential conflict: learning ↔ project progress

- guru (hero) necessary

- potential conflict: guru task ↔ guru's task

- breaking point in thinking about FM:
  before break: FM are a burden
  after break: real benefits
  how did we ever manage without FM?

- not everybody likes FM

Myth 5

FM increase development costs

BOS: probably yes, a little bit

But:

- No real comparison: no non-formal BOS
- Much of costs in learning – first time costs
- More important:

  shift in costs
  from implementation and testing
  towards specification and design

Shift in Efforts

- Design phase more important and much longer

- Gain in implementation and testing

- Design more detailed

- Earlier detection of problems / defects
  "first think and then build"

  In principle, independent from FM
  but without FM easier to escape from it

- Difficult to measure / manage progress
  during design

- Planning of implementation and testing
  very precise

Myth 6

FM are unacceptable to users

BOS: user (RWS) was satisfied

- informal explanations of formal specifications are necessary
- simulation of formal models was very useful to show defects (SPIN with MSC)

Myth 7

FM are not used on real software

BOS: used successfully

And with a next, BOS-like system,
CMG will use FM —
and try to increase the level of formality

> *Dad, when you were young, were there really people who developed their software without formal methods ?*
> *How did they manage ?*

Benefits & Costs

+ better software quality
+ more preciseness
+ problems and defects earlier detected
+ better basis for testing
+ no major problems found during testing
+ better planning of implementation and testing phase
+ efficient reviewing of code based on formal design with uniform style
+ estimated better maintainability (if FM specs kept up to date)

Benefits & Costs

- ◦ design phase longer;
  implementation and testing phase shorter

- ◦ increase in professionalization:
  good projects get better; bad projects get worse

- ◦ likely longer overall development,
  mainly due to learning

- – learning to make effective use of FM not easy

- – integration with current SE practice is weak

- – tool support is insufficient for large systems

Comparison with "Seven Myths"

Observations of Anthony Hall and BOS mostly agree

Minor differences of BOS w.r.t. Hall:

- distinction specification $\leftrightarrow$ design
- distinction specification $\leftrightarrow$ model
- FM not ideal for conceptual modelling and high-level structuring
- importance of tools:
  - learning
  - stimulating
  - check of specifications
  - model checking
  - but insufficient:
    - functionality
    - size which can be handled
    - integration with SE tools

Conclusion FM Usage

- FM do increase quality

- FM are usable in industrial context

- FM require some learning and adaptation, not completely off-the-shelf

Conclusion FM Research

- Not so much need for new FM theories
  but making existing FM theories
  better applicable

- Integration of data and process formalisms

- Better tools needed w.r.t. to size of models

- Integration within SE practice

- Bottle-neck in early development phases

- Proving code correct w.r.t. formal specification
  is not important

- Getting the formal specification is important

More about the Storm Surge Barrier:

http://www.minvenw.nl/rws/dzh/svk/engels/