Model-Driven Instrumentation for Dynamic Analysis of Legacy Software Systems

André van Hoorn\textsuperscript{1}, Holger Knoche\textsuperscript{2}, Wolfgang Goerigk\textsuperscript{2}, and Wilhelm Hasselbring\textsuperscript{1}

\textsuperscript{1}Software Engineering Group, University of Kiel, Germany
\textsuperscript{2}b+m Informatik AG, Melsdorf, Germany

May 03, 2011 @ WSR 2011, Bad Honnef
Dynamic Analysis for Model-Driven Software Modernization

Definition of Transformations

Dynamic Analysis

Static Analysis

Model-Based Testing

Code Generation

Evaluation

v. Hoorn, Knoche, Goerigk, Hasselbring

Model-Driven Instrumentation for Dynamic Analysis
Context — DynaMod Project

Dynamic Analysis for Model-Driven Software Modernization

Project Consortium:

1. **b+m Informatik AG**
   *(Development partner, consortium leader)*
   - Comprehensive MDSD know-how
   - Initiated openArchitectureWare (oAW)

### Additional Information

- **b+m Informatik AG**, Melsdorf
  - [http://www.bmiag.de/](http://www.bmiag.de/)

- **Software Engineering Group**, University of Kiel, Kiel
  - [http://se.informatik.uni-kiel.de/](http://se.informatik.uni-kiel.de/)

- **Dataport**, Altenholz
  - [http://www.dataport.de/](http://www.dataport.de/)

- **HSH Nordbank AG**, Kiel
  - [http://www.hsh-nordbank.de/](http://www.hsh-nordbank.de/)
Context — DynaMod Project

Dynamic Analysis for Model-Driven Software Modernization

Project Consortium:

1. **b+m Informatik AG**
   (Development partner, consortium leader)
   - Comprehensive MDSD know-how
   - Initiated openArchitectureWare (oAW)

2. **Software Engineering Group, Univ. Kiel**
   (Research partner)
   - Model-driven engineering, operation, and evolution of software systems
   - Emphasis on software quality (of service)
Context — DynaMod Project

Context & Motivation

DynaMod

Dynamic Analysis for Model-Driven Software Modernization

Project Consortium:

1. **b+m Informatik AG**
   *Development partner, consortium leader*
   - Comprehensive MDSD know-how
   - Initiated openArchitectureWare (oAW)

2. **Software Engineering Group, Univ. Kiel**
   *Research partner*
   - Model-driven engineering, operation, and evolution of software systems
   - Emphasis on software quality (of service)

3. **Dataport**
   *Associated partner*
   - Provides ICT services for public/tax administrations
**Context — DynaMod Project**

**Context & Motivation**

DynaMod

Dynamic Analysis for Model-Driven Software Modernization

**Project Consortium:**

1. **b+m Informatik AG**  
   *(Development partner, consortium leader)*
   - Comprehensive MDSD know-how
   - Initiated openArchitectureWare (oAW)

2. **Software Engineering Group, Univ. Kiel**  
   *(Research partner)*
   - Model-driven engineering, operation, and evolution of software systems
   - Emphasis on software quality (of service)

3. **Dataport**  
   *(Associated partner)*
   - Provides ICT services for public/tax administrations

4. **HSH Nordbank AG**  
   *(Associated partner)*
   - Leading bank for corporate and private clients in northern Germany

---

v. Hoorn, Knoche, Goerigk, Hasselbring  
Model-Driven Instrumentation for Dynamic Analysis  
May 03, 2011  
2 / 15
DynoMod

Dynamic Analysis for Model-Driven Software Modernization

Project Consortium:

1. b+m Informatik AG
   (Development partner, consortium leader)

2. Software Engineering Group, Univ. Kiel
   (Research partner)

3. Dataport
   (Associated partner)

4. HSH Nordbank AG
   (Associated partner)

Funding:

- BMBF “KMU-innovativ”
- 2 years (01/11–12/12)

Read More:

- [http://kosse-sh.de/dynamod/](http://kosse-sh.de/dynamod/)
- MDSM @ CSMR 2011 [vHFG+11]
Dynamic Analysis for Model-Driven Software Modernization

DynaMod

Definition of Transformations

Dynamic Analysis

Code Generation

Model-Based Testing

Static Analysis

Evaluation

v. Hoorn, Knoche, Goerigk, Hasselbring

Model-Driven Instrumentation for Dynamic Analysis

May 03, 2011
Motivation

Static analysis is not sufficient to study a system’s architecture comprehensively.
Motivation

Static analysis is not sufficient to study a system’s architecture comprehensively.

How to Gather Runtime Data from Executing Systems? — Instrumentation

- Profiling — employed in development environments; considerable performance overhead
- Monitoring — employed in production environments; captures real usage profile
Motivation

Static analysis is not sufficient to study a system’s architecture comprehensively.

How to Gather Runtime Data from Executing Systems? — Instrumentation

- **Profiling** — employed in development environments; considerable performance overhead
- **Monitoring** — employed in production environments; captures real usage profile

Use Cases — Online & Offline Analysis

The obtained monitoring data can, for instance, be used for

- **Performance evaluation** (e.g., bottleneck detection)
- **(Self-)adaptation control** (e.g., capacity management)
- **Application-level failure detection and diagnosis**
- **Service-level management**
Dynamic Analysis of Software Runtime Behavior

Context & Motivation

Motivation

Static analysis is not sufficient to study a system’s architecture comprehensively.

How to Gather Runtime Data from Executing Systems? — Instrumentation

- **Profiling** — employed in development environments; considerable performance overhead
- **Monitoring** — employed in production environments; captures real usage profile

Use Cases — Online & Offline Analysis

The obtained monitoring data can, for instance, be used for

- **Performance evaluation** (e.g., bottleneck detection)
- **(Self-)adaptation control** (e.g., capacity management)
- **Application-level failure detection and diagnosis**
- **Service-level management**
- **Software reengineering**: (architecture) reconstruction, modernization
Agenda

1. Context & Motivation
2. Model-Driven Instrumentation for Dynamic Analysis
3. AOP for Legacy Languages
4. Monitoring Instrumentation
5. Conclusions
1. Context & Motivation

2. Model-Driven Instrumentation for Dynamic Analysis

3. AOP for Legacy Languages

4. Monitoring Instrumentation

5. Conclusions
Model-Driven Instrumentation for Dynamic Analysis

Overview

Domain

Architecture

Implementation

v. Hoorn, Knoche, Goerigk, Hasselbring

Model-Driven Instrumentation for Dynamic Analysis

May 03, 2011 6 / 15
Overview

Model-Driven Instrumentation for Dynamic Analysis

- **Domain**
- **Architecture**
- **Implementation**

- DSL
- ADL
- Shop
- search()
- AST
- Code
- Instr. code

Model-Driven Software Development
Model-Driven Instrumentation for Dynamic Analysis

Overview

Model Driven Software Development

- Static Analysis
- Model-Driven Software Development
- Implementation
  - AST
  - Code
  - Instr. code
- Architecture
  - <<RespTReq>>
  - <<AvgRespT>>
  - <<OpExecProbe>>
- Domain
  - Queries
  - Analysis
  - Measures
  - Instrumentation
  - Directives

AST
DSL
ADL
Shop
search()
Instr.
code
Overview

Model-Driven Instrumentation for Dynamic Analysis

- Queries
- Analysis Measures
- Instrumentation Directives

- DSL
- ADL
- Shop
- search()
- Code
- Instr. code

- Static Analysis
- Model-Driven Software Development
- AST

Domain

Architecture

Implementation

- <<RespTReq>>
- <<AvgRespT>>
- <<OpExecProbe>>
Overview

Model-Driven Instrumentation for Dynamic Analysis

Architect...
Model-Driven Instrumentation for Dynamic Analysis

- Queries
- Analysis Measures
- Instrumentation Directives
- DSL
- ADL
- Shop
- search()
- Instr.
- code

- Domain
- Architecture
- Implementation
- Model-Driven Instrumentation
- Model-Driven Software Development
- Static Analysis
- Dynamic Analysis

- Monitoring Events
- Results
- Measurement
- :AvgRT
  val=730
- :EvalRes
  val=true
- :OpExec
  tin=211
tout=955

- v. Hoorn, Knoche, Goerigk, Hasselbring

May 03, 2011
Overview—DynaMod Examples
Public Sub searchBook()
...
crm.getOffers
End Sub

Public Sub searchBook()
...
crm.getOffers
End Sub

Code
Public Sub searchBook()
...
crm.getOffers
End Sub


crm.getOffers
Public Sub searchBook()
...
crm.getOffers
End Sub
Overview—DynaMod Examples

Model-Driven Instrumentation for Dynamic Analysis

Public Sub searchBook()
...
crm.getOffers
End Sub

DMeasures
: AvgOperationRTMeasure
: OperationInvocationCountMeasure

DADL (DynaMod ADL)
Catalog
Bookstore
searchBook()

Sub
SIGNATURE
Public
searchBook
PARAMETERS
STATEMENTS

AST

DMeasures

Code
Public Sub searchBook()
...
crm.getOffers
End Sub

DMeasures
: AvgOperationRTMeasure
: OperationInvocationCountMeasure

DInstrumentation
: OperationExecutionProbe

DADL (DynaMod ADL)

Catalog
Bookstore
searchBook()
Overview—DynaMod Examples

Model-Driven Instrumentation for Dynamic Analysis

Public Sub searchBook()
...
crm.getOffers
End Sub

Code

v. Hoorn, Knoche, Goerigk, Hasselbring

Model-Driven Instrumentation for Dynamic Analysis

May 03, 2011
Model-Driven Instrumentation for Dynamic Analysis

Overview—DynaMod Examples

DMeasures

:AvgOperationRTMeasure

:OperationInvocationCountMeasure

DInstrumentation

:OperationExecutionProbe

DADL (DynaMod ADL)

Catalog

Bookstore

searchBook()

Public Sub

searchBook()
...
crm.getOffers
End Sub

Instrumented Code

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

crm.getOffers
End Sub
Public Sub searchBook()
...
crm.getOffers
End Sub

@intercept#Call:OpCallIcptr["Bookstore", "searchBook"]

DADL (DynaMod ADL)

Instrumented Code

MonitoringEvents

Public Sub searchBook()
...
crm.getOffers
End Sub

(@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

Measures

OperationInvocationCountMeasure

AverageOperationRTMeasure

Instrumentation

OperationExecutionProbe

Catalog

Bookstore

searchBook()
Public Sub searchBook()
...
crm.getOffers
End Sub
Public Sub searchBook()
...
crm.getOffers
End Sub

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook"]

Public Sub searchBook()
...
crm.getOffers
End Sub

'@intercept#Execution:OpExecIcptr["Bookstore", "searchBook"]

DADL (DynaMod ADL)

DMeasures
:AvgOperationRTMeasure

:OperationInvocationCountMeasure

DInstrumentation
:OperationExecutionProbe

Catalog
Bookstore

searchBook()

DADL (DynaMod ADL)

DMeasures
:AvgOperationRT

avgRTMillis=5676

:OperationInvocationCount

count=67643

Instrumented
MonitoringRecord

Monitoring
Events

:OperationExecution

tin=34
tout=38

class="CRM"
operation="getOffers"

:OperationExecutionRecord

tin=34
tout=38

class="CRM"
operation="getOffers"

DEvent
Monitoring
Events

:OperationExecution

tin=34
tout=38

class="CRM"
operation="getOffers"

:OperationExecutionRecord

tin=34
tout=38

class="CRM"
operation="getOffers"
Agenda

1. Context & Motivation
2. Model-Driven Instrumentation for Dynamic Analysis
3. AOP for Legacy Languages
4. Monitoring Instrumentation
5. Conclusions
Public Sub searchBook()
...
  crm.getOffers
End Sub
Public Sub searchBook()
'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]
crm.getOffers
End Sub
Public Sub searchBook()
...
crm.getOffers
End Sub

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

OpCallIcptr
before(IcptOpCall)
after(IcptOpCall)
Public Sub searchBook()
...
crm.getOffers
End Sub

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

OpCallIcptr
before(IcptOpCall)
after(IcptOpCall)

IcptOpCall
callerClass:String
callerMethod:String
calleeClass:String
calleeMethod:String
object:Variant
Public Sub searchBook()
...
crm.getOffers
End Sub

Private Sub Class_Initialize()
...
End Sub

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", 
"CRM", "getOffers"]
AOP Framework—VB6 Example

AOP for Legacy Languages

Public Sub searchBook()
...
crm.getOffers
End Sub

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

Private Sub Class_Initialize()
...
End Sub

'@intercept#Execution:OpExecIcptr["Bookstore", "Class_Initialize"]
AOP Framework—VB6 Example

AOP for Legacy Languages

```vbscript
Public Sub searchBook()
    ... 
crm.getOffers
End Sub

Private Sub Class_Initialize()
    ...
End Sub

'@intercept#Execution:OpExecIcptr["Bookstore", "Class_Initialize"]

OpExecIcptr
    before(IcptOpExec)
    after(IcptOpExec)

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

OpCallIcptr
    before(IcptOpCall)
    after(IcptOpCall)
```

callerClass:String
callerMethod:String
calleeClass:String
calleeMethod:String
object:Variant

v. Hoorn, Knoche, Goerigk, Hasselbring
Model-Driven Instrumentation for Dynamic Analysis
May 03, 2011
AOP Framework—VB6 Example

AOP for Legacy Languages

Public Sub searchBook()
...
crm.getOffers
End Sub

'@intercept#Call:OpCallIcptr["Bookstore", "searchBook", "CRM", "getOffers"]

Private Sub Class_Initialize()
...
End Sub

'@intercept#Execution:OpExecIcptr["Bookstore", "Class_Initialize"]

OpExecIcptr
before(IcptOpExec)
after(IcptOpExec)

OpCallIcptr
before(IcptOpCall)
after(IcptOpCall)

IcptOpCall
callerClass:String
callerMethod:String
calleeClass:String
calleeMethod:String
object:Variant

IcptOpExec
class: String
method: String
object: String

v. Hoorn, Knoche, Goerigk, Hasselbring
May 03, 2011 9 / 15
Public Sub searchBook()
...
Dim iCall0 As IcptOpCall
Set iCall0 = New IcptOpCall
Call iCall0.init("Bookstore", "searchBook", "CRM", "getOffers")
Call opCallIntcptr.before(iCall0)
crm.getOffers
Call opCallIntcptr.after(iCall0)
End Sub
Model-Driven Instrumentation for Dynamic Analysis
AVB6C — IDE Integration

AOP for Legacy Languages

Select main project
Select aspect project
Select output path

Projects

Weaver options
Logging options

*overwrite output files
*add info-marks
*copy all directories
*accept hidden files
*accept files outside base directories
*cleanup on error
*compile after weaving

Exclude files from process
List of files/directories to be excluded from weaving/compiling, separated by semicolon:
*.scc ; myTextFile.txt ; "another file.jpg"

Logging messages

Weaving file "home/avanhoornSVN_work/dynamics/trunk/src/AspectVB6/AspectVB6-examples/bookstore-vb6-annotated/Bookstores.vbp"
Transforming (Line 27): @intercept#Execution:OpExecOpExpr1("Bookstore","Class_Initialize")
Transforming (Line 42): @intercept#CallOpCallOpExpr1("Bookstore","searchBook","Catalog","
Transforming (Line 50): @intercept#CallOpCallOpExpr1("Bookstore","searchBook","CRM","get
Weaving file "home/avanhoornSVN_work/dynamics/trunk/src/AspectVB6/AspectVB6-examples/bookstore-vb6-annotated/Bookstores.vbp"
Weaving file "home/avanhoorn/SVN_work/dynamics/trunk/src/AspectVB6/AspectVB6-examples/bookstore-vb6-annotated/Bookstores.vbp"
Weaving file "home/avanhoorn/SVN_work/dynamics/trunk/src/AspectVB6/AspectVB6-examples/bookstore-vb6-annotated/Bookstores.vbp"
Inserting global declarations
0 directories created, 4 files woven, 7 files copied, 1 new files created,
Output project: "home/avanhoorn/SVN_work/dynamics/trunk/src/AspectVB6/tmp/Bookstores.vbp"
Finished.
AVB6C — IDE Integration

AOP for Legacy Languages

V. Hoorn, Knoche, Goerigk, Hasselbring

Model-Driven Instrumentation for Dynamic Analysis

May 03, 2011
Acknowledgment: Our DynaMod student assistants, **Eike Schulz and Benjamin Schnoor**, are responsible for large parts of the implementation!
1. Context & Motivation

2. Model-Driven Instrumentation for Dynamic Analysis

3. AOP for Legacy Languages

4. Monitoring Instrumentation

5. Conclusions
**Kieker Monitoring & Analysis Framework**

Monitoring Instrumentation

- **Monitoring Instrumentation**
  - Monitoring log
    - e.g., file system, database, message-oriented middleware

- **Analysis Plug-In**
  - e.g., trace information, workload, response times, resource utilization, loop counts

- **Monitoring Record**
  - e.g., AOP-based method call interception

- **Analysis Plug-In**
  - e.g., architecture reconstruction, performance evaluation, online adaptation control, failure diagnosis

---

**Kieker Framework — Core Characteristics [vHRH+09]**

- **Flexible architecture** (custom *probes, readers, writers, analysis plug-ins*)
- **Integrated & extensible record type model** for monitoring & analysis
- **Logging, reconstruction, analysis/visualization of (distributed) traces**
- **Low overhead** (designed for continuous operation in multi-user systems)
- **Evaluated in industry case studies**

---

http://kieker.sourceforge.net

---

v. Hoorn, Knoche, Goerigk, Hasselbring

Model-Driven Instrumentation for Dynamic Analysis

May 03, 2011 12 / 15
Kieker Monitoring & Analysis Framework

Flexible architecture (custom probes, readers, writers, analysis plug-ins)

Integrated & extensible record type model for monitoring & analysis

Logging, reconstruction, analysis/visualization of (distributed) traces

Low overhead (designed for continuous operation in multi-user systems)

Evaluated in industry case studies

Kieker Framework — Core Characteristics [vHRH+09]

http://kieker.sourceforge.net
• Goal: Reuse existing Java-based Kieker.Monitoring component
• Goal: Reuse existing Java-based Kieker.Monitoring component
• Requires communication bridge among legacy language and Java
• Goal: Reuse existing Java-based Kieker.Monitoring component
• Requires communication bridge among legacy language and Java
• Instantiation of presented AOP framework for monitoring probes
Instrumentation of Legacy Languages

Monitoring Instrumentation

- Goal: Reuse existing Java-based Kieker.Monitoring component
- Requires communication bridge among legacy language and Java
- Instantiation of presented AOP framework for monitoring probes
- Developed proof-of-concept implementation for VB6
• Goal: Reuse existing Java-based Kieker.Monitoring component
• Requires communication bridge among legacy language and Java
• Instantiation of presented AOP framework for monitoring probes
• Developed proof-of-concept implementation for VB6
1 Context & Motivation

2 Model-Driven Instrumentation for Dynamic Analysis

3 AOP for Legacy Languages

4 Monitoring Instrumentation

5 Conclusions
Conclusions

Summary

- Context: DynaMod project for model-driven modernization
- Model-driven instrumentation for dynamic analysis
- AOP for legacy languages
- Monitoring instrumentation employing AOP framework and Kieker
- Current focus on Visual Basic 6
Summary

- Context: DynaMod project for model-driven modernization
- Model-driven instrumentation for dynamic analysis
- AOP for legacy languages
- Monitoring instrumentation employing AOP framework and Kieker
- Current focus on Visual Basic 6

Future Work

- Refine and extend meta-models for queries, measures, instrumentation etc.
- Development of corresponding tool support — e.g., M2M/M2Code transformations
- Evaluate approach on other abstraction layers — e.g., DSLs, ASTs
Conclusions

Summary

- Context: DynaMod project for model-driven modernization
- Model-driven instrumentation for dynamic analysis
- AOP for legacy languages
- Monitoring instrumentation employing AOP framework and Kieker
- Current focus on Visual Basic 6

Future Work

- Refine and extend meta-models for queries, measures, instrumentation etc.
- Development of corresponding tool support — e.g., M2M/M2Code transformations
- Evaluate approach on other abstraction layers — e.g., DSLs, ASTs
- Additional AOP features — e.g., loops, branches
- Additional programming languages — e.g., COBOL, Natural, Structured Text
- Study performance overhead
Thomas Stahl and Markus Völter.

DynaMod project: Dynamic analysis for model-driven software modernization.

André van Hoorn, Holger Knoche, Wolfgang Goerigk, and Wilhelm Hasselbring.
Model-driven instrumentation for dynamic analysis of legacy software systems.

André van Hoorn, Matthias Rohr, Wilhelm Hasselbring, Jan Waller, Jens Ehlers, Sören Frey, and Dennis Kieselhorst.
Continuous monitoring of software services: Design and application of the Kieker framework.
Technical Report TR-0921, Department of Computer Science, University of Kiel, Germany, November 2009.