TQL: A Query Language to Support Traceability

Software Development Laboratory
<SDML>

Jonathan I. Maletic

Michael L. Collard

KENT STATE UNIVERSITY

<SDML>

Artifact Representation

- Models are represented in XML

- Any structured/semi-structured type of model is supported

Artifact	Model
Source code	srcML [Maletic et al. 02]
UML design documents	classML …
Requirements	UseCaseML …
:	:

KENT STATE UNIVERSITY

<SDML>

Link Representation

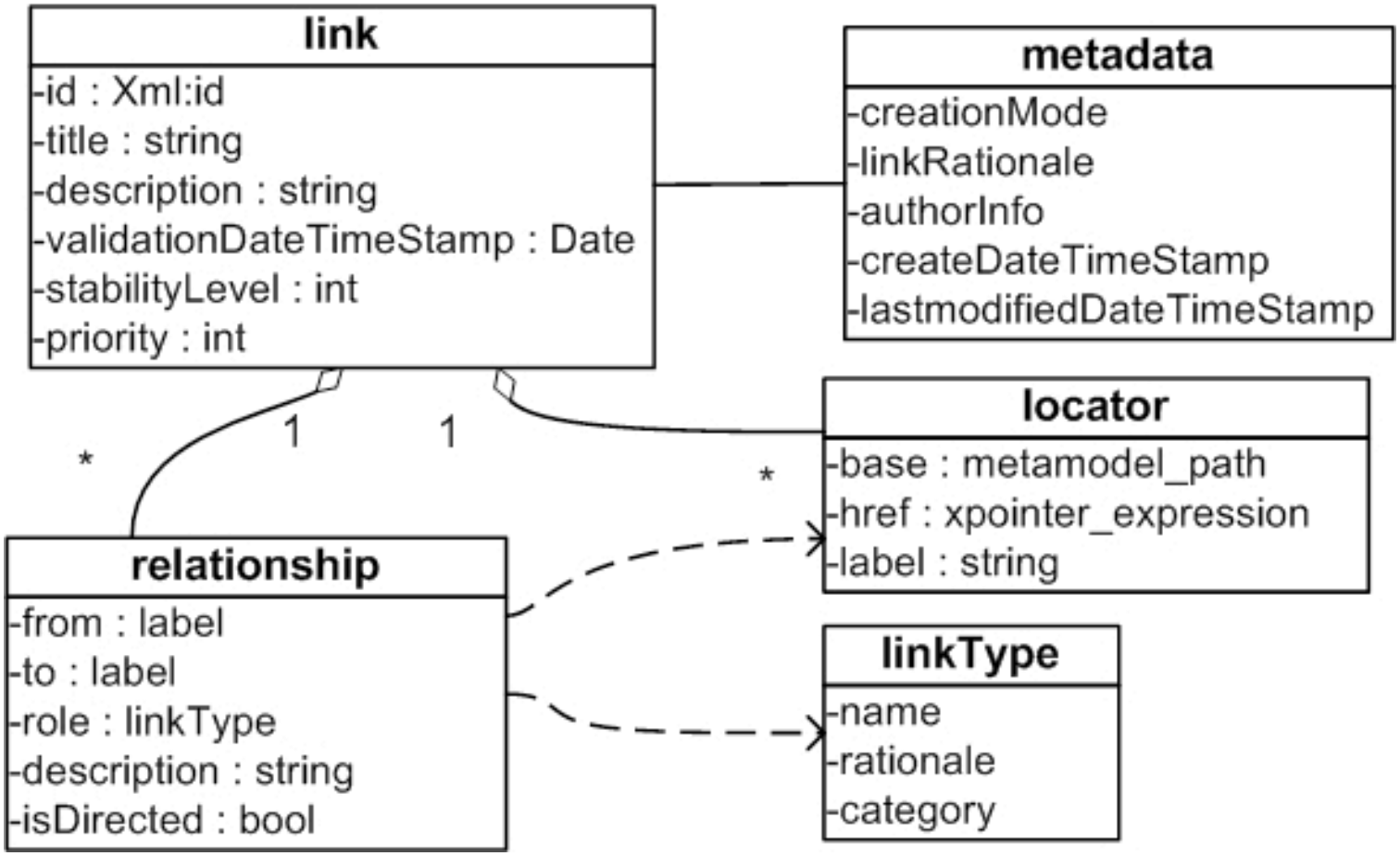
- XPath forms address paths between elements of models

- Links are stored external to models
  - Original model is preserved
  - Multiple views are possible

KENT STATE UNIVERSITY

<SDML>

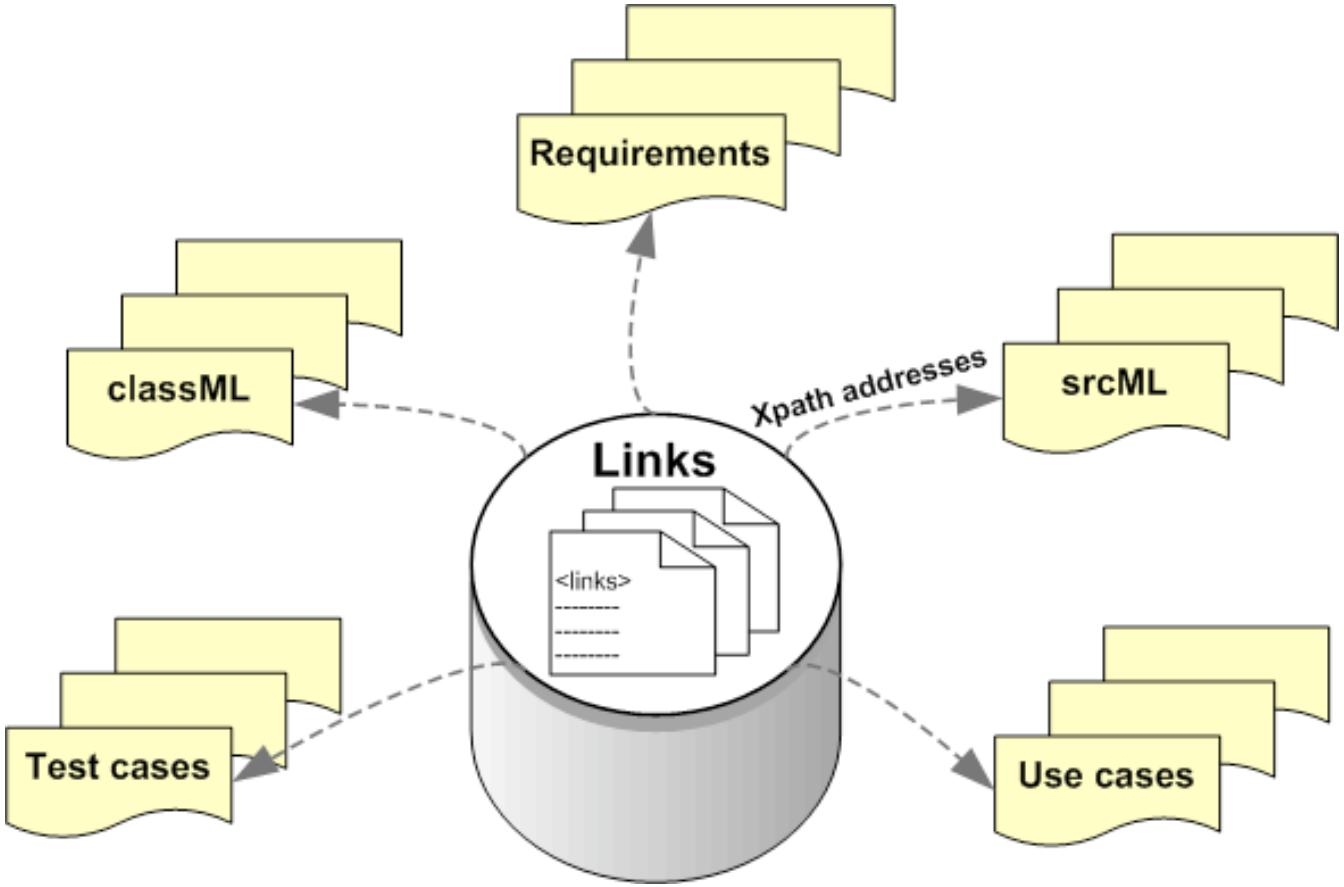
Traceability Link Model



KENT STATE UNIVERSITY

<SDML>

Link Storage and Addressing



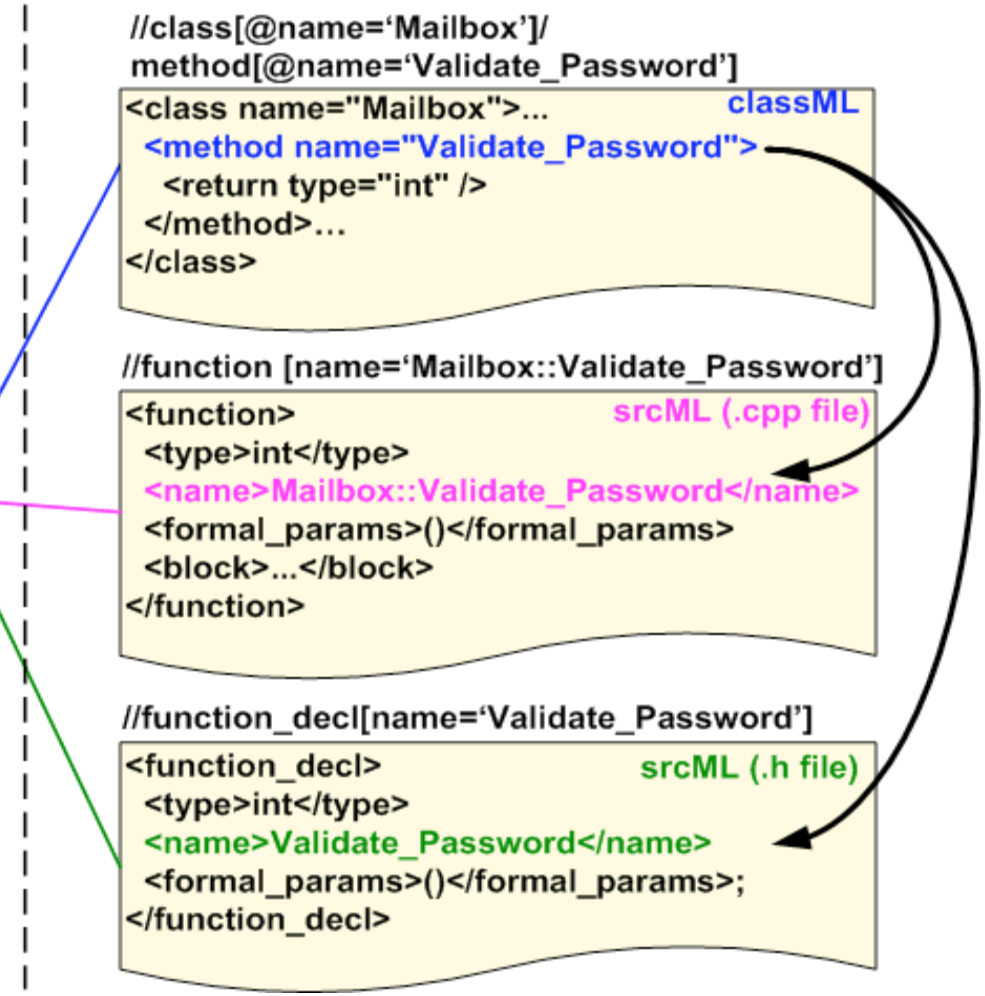
KENT STATE UNIVERSITY

<SDML>

Sample Link

Sample Link Record

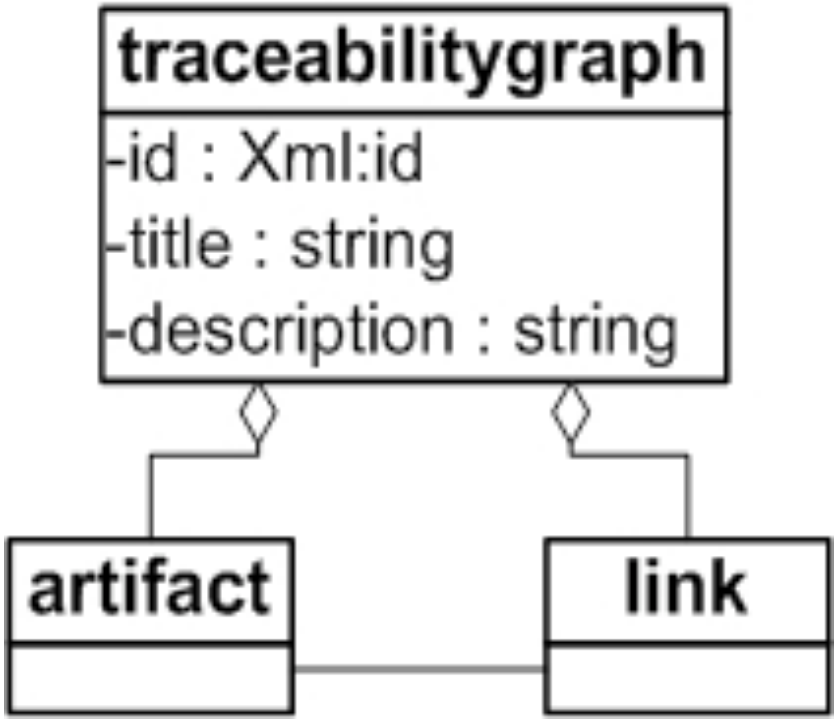
id	myex1
stability	1
priority	1
valid-date	1/14/2007 12:43:20EDT
title	method name must match
metadata	ex1.dcmi.xml
from-locator	Mailbox::Validate_Password
	in classML
to-locator	Mailbox::Validate_Password
	in srcML (.cpp file)
to-locator	Mailbox::Validate_Password
	in srcML (.h file)
arity	n-ary, unidirectional
directed	yes
role	must-agree



KENT STATE UNIVERSITY

<SDML>

Traceability Graph



KENT STATE UNIVERSITY

<SDML>

Artifact Primitives

- art:UseCase()
- art:Requirement()
- art:Design()
- art:Code()
- art:TestCase()
- art:BugReport()

KENT STATE UNIVERSITY

<SDML>

Traceability Primitives

- **`tql:traceTo(`*`source, sink`*`)`**
  - Returns all the artifacts in the traceability graph in the set source that trace directly or indirectly to an artifact in the set sink.

- **`tql:traceFrom(`*`sink, source`*`)`**
  - Returns all the artifacts in the traceability graph in the set *sink* that trace directly or indirectly from an artifact in the set *source*.

- Computes transitive closure

KENT STATE UNIVERSITY

<SDML>

Link Primitives

- **`tql:link(source, sink)`**
  - Transitive closure of all links between the two sets of artifacts
- **`tql:directlink(source, sink)`**
  - Just directly linked artifacts
- **`tql:link(artifact)`**
  - All links to or from an artifact
- **`tql:artifact(links)`**
  - All artifacts involved in a given set of links

KENT STATE UNIVERSITY

<SDML>

Are all requirements covered by a test case?

```
set:difference(
        Requirement(),
        traceFrom(Requirement(),
                  TestCase()    ))
```

KENT STATE UNIVERSITY

<SDML>

Are all non-functional requirements addressed by one or more parts of the implementation?

```
tql:traceTo(
    Requirement()[NFR()],
    Code())

tql:traceTo(
    Requirement()[NFR()],
    tql:traceTo(TestCase()
                Code()) )
```

KENT STATE UNIVERSITY

<SDML>

Which parts of code covered by a requirement have documented pre and post condition?

```
tql:traceFrom(
    Requirement(R),
    Code()//src:function
          [@requires | @ensures])
```

KENT STATE UNIVERSITY

<SDML>

Are any test cases missing?

```
set:difference(
    Requirement(),
    tql:traceTo(
            Requirement(),
            TestCase()))
```

KENT STATE UNIVERSITY

<SDML>

What is the impact of changing a requirement on the safety of the system?

```
tql:impactAnalysis(
  tql:traceFrom(
    Code(),
    Tql:traceFrom(
                Design(),
                Requirement(R))))
```

KENT STATE UNIVERSITY

<SDML>

Related Work

- Constraint based mechanism to indicate when artifacts are inconsistent with one another [Reiss 2002, 2005, 2006]
- Consistency Checkers - xLinkit [Nentwich et al. 2002]
- Policy centric approach - ArchTrace [Murta et al. 2006]

KENT STATE UNIVERSITY

<SDML>

Conclusions & Future

- Working on implementation – release as a tool
- Working on other primitives
- Applying to commercial software system

- This work is funded in part by the National Science Foundation under NSF grant CCF 08-11-21.