

RE II: Traceability & Requirements Evolution

Traceability and Software Execution

Group 1 :

Michael Ammann (presenting)

Katja Kevic (presenting)

Nicolas Hoby

Thomas Ritter

Introduction (1)

- Ideal process shall involve:
 - Incoming change request
 - Analyze the impact of the change
 - Update of requirements specification
 - Modify implementation
- But: Usually not the case

Introduction (2)

- Why is it important to have an up-to-date requirements specification?
 - Facilitate program comprehension
 - Provide the rationale behind the implementation
 - Requirements specification is accessible to all stakeholders
 - Facilitates the application of well-considered changes

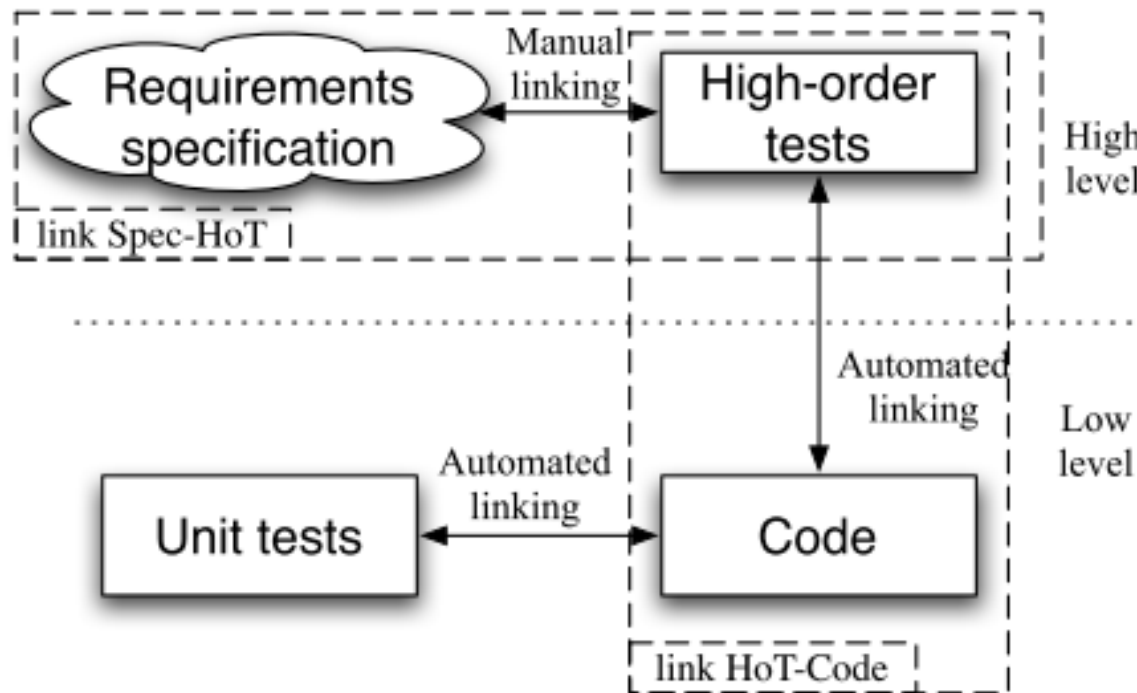
Approaches

- Manual update of requirements specification
 - Time consuming, error-prone, complex
- (Semi-) Automated approaches:
 - Hint Generation approach
 - Trace Analyzer

Tests to link requirements and implementation

- Test are usually maintained with the implementation
- Tests, which are meant to check the external behavior of the system, are derived from requirements

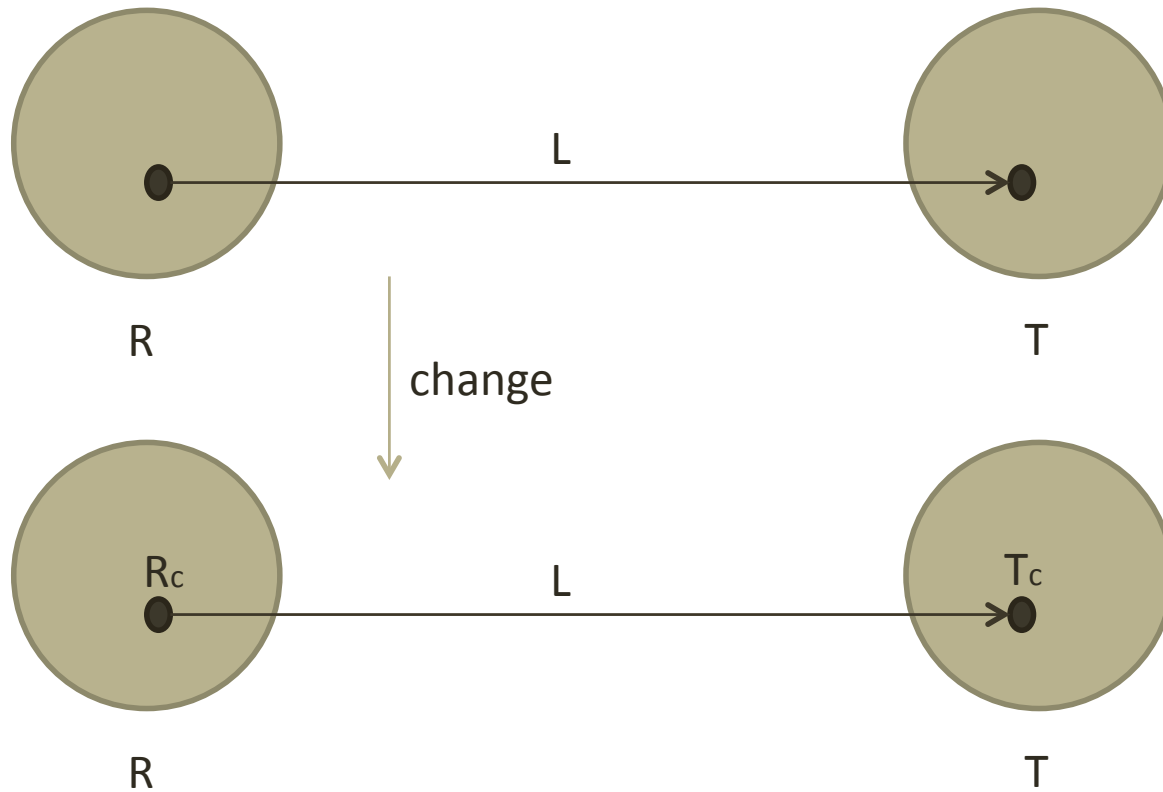
Basic Idea



- Existence of a well-maintained high-order test suite with good requirements coverage
- Existence of dependable traceability links between requirements and high-order tests

Hint Generation (1)

- Rules



Hint Generation (2)

- Likelihood of a requirement to be affected
 - 5: The requirement is affected by the change
 - 4: The requirement is likely to be affected
 - 3: We cannot decide about the requirement
 - 2: The requirement is likely not to be affected by the change
 - 1: The requirement is not affected by the change
- Rule A,B,C,D

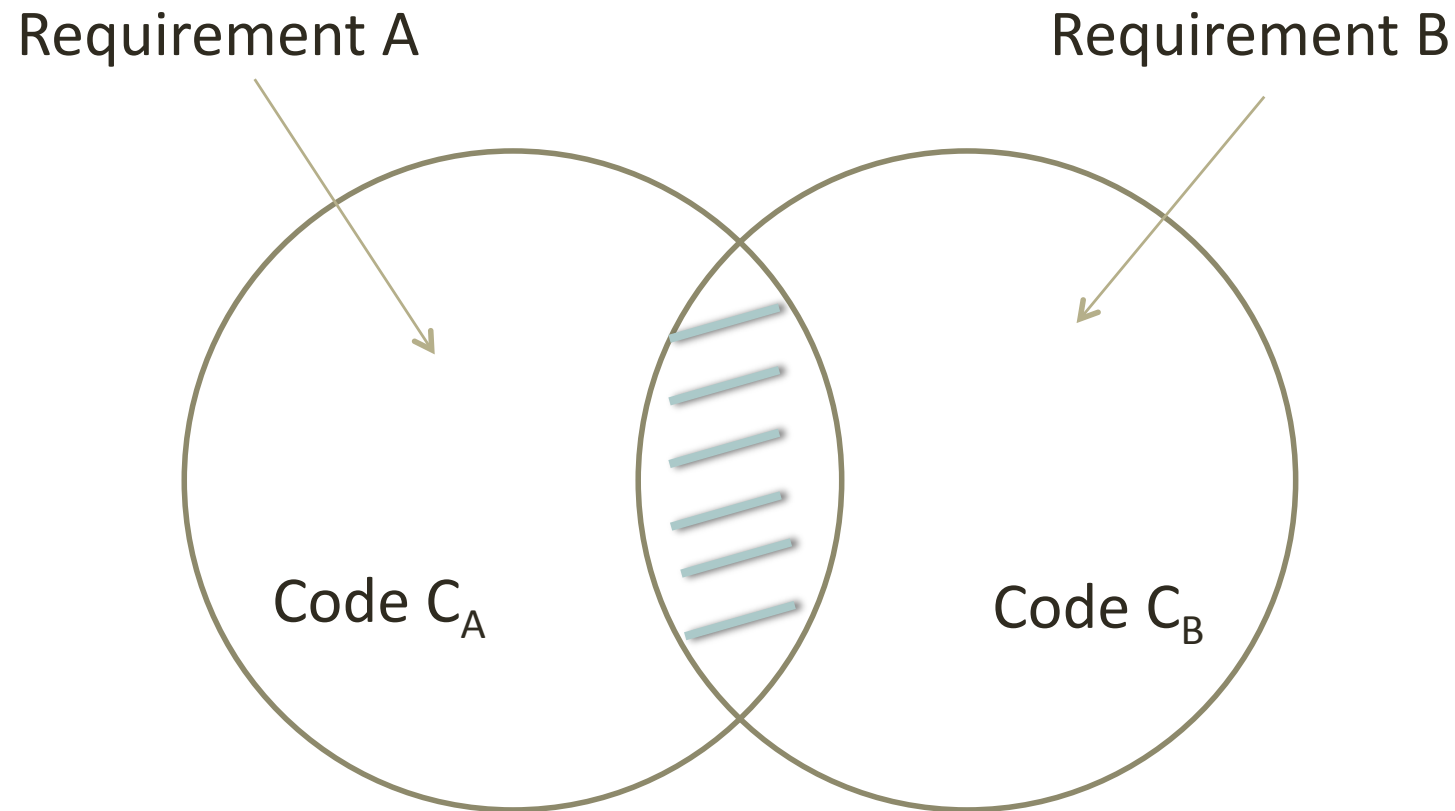
Hint Generation about new requirements

- Information about new requirements directly from the source code
- Analyzing of test execution traces
 - Selective tracing
- Further hint: relation between new and old requirements

Trace Analyzer

- Defines trace dependencies through:
 - Transitive reasoning
 - «Common ground»
- Iterative process:
 - Build graph
 - Nodes = common grounds and their overlaps
 - Edges = dependencies
 - Manipulate nodes
 - Establish trace dependencies

Common Ground – Source Code



List of Requirements

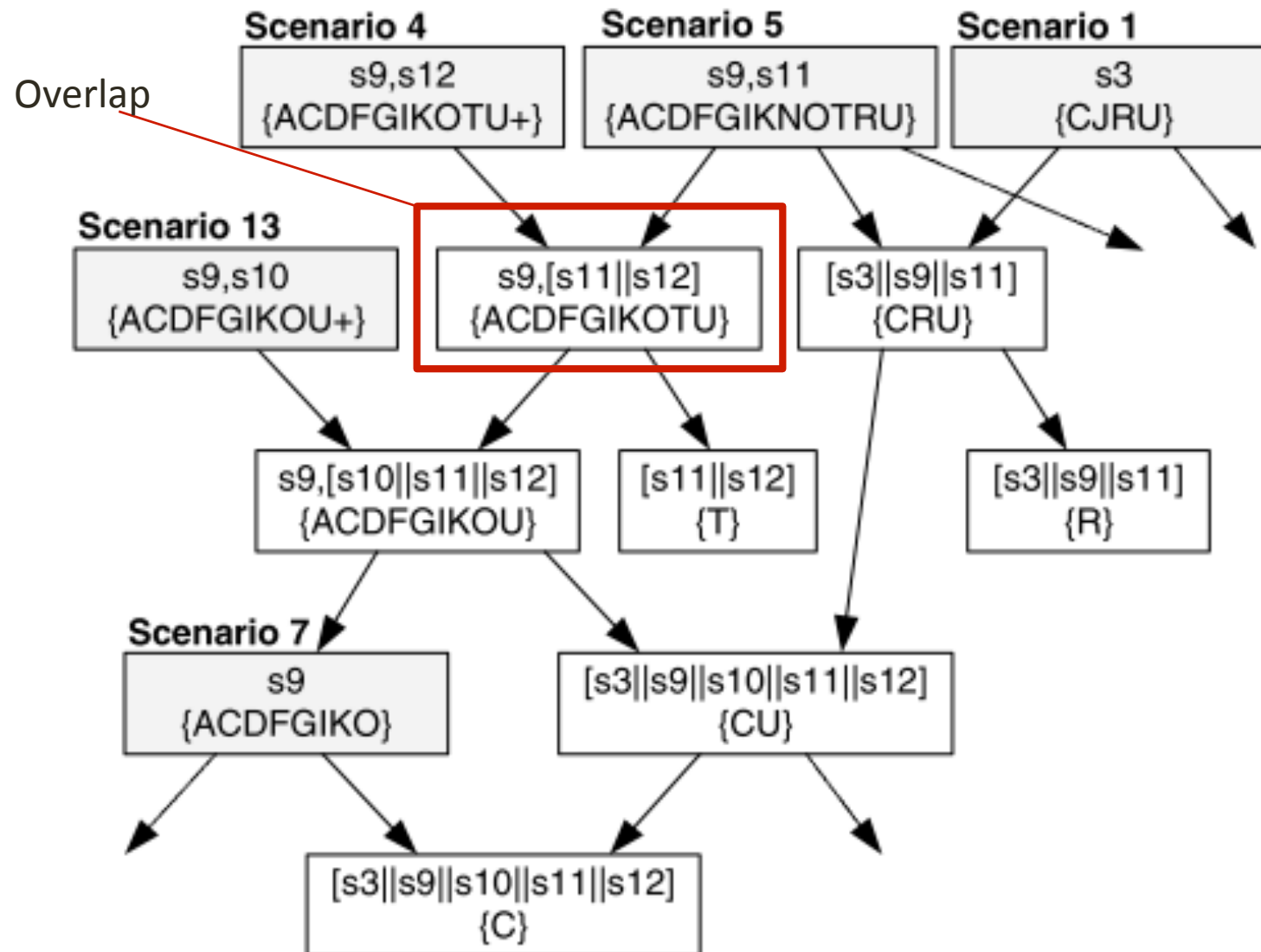
r0	Users should be able to display a list of available movies and select one from the list
r1	Play movie immediately after selection from list
r2	Users should be able to display textual information about a selected movie
r3	User should be able to pause a movie
r4	3 seconds max to load movie list
r5	3 seconds max to load textual information about a movie
r6	1 second max to start playing a movie
r7	Provide VCR-like user interface
r8	User should be able to stop a movie
r9	User should be able to start a movie

Scenarios and Observed Footprints

Common Ground

<i>Test Scenario</i>	<i>Artifact</i>	<i>Observed Java Classes</i>
1. view movie list	[s3]	[C,J,R,U]
2. view textual movie information	[s4,s6][r2]	[C,E,J,N,R]
3. select/play movie	[s8,s9][r6]	[A,C,D,F,G,I,J,K,N,O,T,R,U]
4. press stop button	[s9,s12][r8]	[A,C,D,F,G,I,K,O,T,U]
5. press play button	[s9,s11][r9]	[A,C,D,F,G,I,K,N,O,T,R,U]
6. change server	[s5,s7]	[C,R,J,S]
7. playing	[s9]	[A,C,D,F,G,I,K,O]
8. get textual movie information	[r5]	[N,R]
9. movie list	[r4]	[R]
10. VCR-like UI	[r7]	[A,C,D,F,G,I,K,N,O,R,T,U]
11. select movie	[r0]	[C,J,N,R,T,U]
12. select/play movie	[r1]	[A,C,D,F,G,I,J,K,N,O,R,T,U]
13. press pause	[s9,s10][r3]	[A,C,D,F,G,I,K,O,U]

Footprint Graph



Artifacts to Java Class Dependencies

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
r0			F							F				F				F		F	F
r1	F		F	F		F	F		F	F	F			F	F			F		F	F
r2			F		F					F				F				F			
r3	F		F	F		F	F		F		F				F						F
r4																		F			
r5														F				F			
r6	F		F	F		F	F		F	F	F			F	F			F		F	F
r7	F		F	F		F	F		F		F			F	F			F		F	F
r8	F		F	F		F	F		F		F				F					F	F
r9	F		F	F		F	F		F		F			F	F			F		F	F
s3			P							P								P			P
s9	F		P	F		F	F		F		F				F						
s10			P																		P
s11			P											P				P		P	P
s12			P																	P	P

Advantages / Drawbacks

- + (Semi-) automatable and tool supported
- + Creates non-obvious dependencies
- + Understanding requirements origins
- + Identification of missing requirements
- + Determination of change impact
- Requires manual input

Conclusion

- Important to have up-to-date requirements specification
- Automated approaches are less
 - Time consuming, error-prone, complex
- Hint Generation based on tests
- Trace Analyzer for generating trace information automatically