L7 – Traceability

DAT232/DIT285 Advanced Requirements Engineering

Eric Knauss
eric.knauss@cse.gu.se


CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

September 30, 2025

Visit **gosocrative.com** and enter room name REQENG

Lecture starts at 13:15

You are welcome to share a short break with us and discuss/ask questions

Outline

Housekeeping

- R1 feedback sent. Discuss with supervisors and also during workshop this week.
- WS4 (ICC): Let's investigate how your group is working and finetune towards R3
- Idea behind 3 releases: Do, assess, feedback, reflect, do better... we will only grade at R3. R1 and R2 are passed when submitted (and very little reason to do rework).

Quick recap (prioritization): → `socrative.com`, Room REQENG, Q1

Feedback from Socrative

7.

Please name one thing that you liked and one thing that you wished for in this lecture.

Hide Answers | Show Names | 2/39 Students Answered

dont knwo

i would like to have the slides few days before the course like have them in the weekend for the next week's lecture

Sync of lectures and project, 2024

8. How well are the lectures (so far) aligned to your progress in the project?

Hide Results 3/51 Students Answered

A	The lecture provide knowledge **much too early** for the project (i.e., the project lags >2 weeks behind the lectures)	33%
B	The lectures provide knowledge **too early** for the project (i.e., the project lags >1 week behind the lectures)	33%
C	The lectures provide knowledge at ***roughly* the right moment** in the project (i.e., +/- 1 week)	0%
D	The lectures provide the knowledge **too late** for the project (i.e., the lectures lag >1 week behind the project)	33%
E	The lectures provide the knowledge **much too late** for the project (i.e., the lectures lag >2 week behind the project)	0%

9. In what degree does the lecture material help you in the project?

Hide Results 3/51 Students Answered

A	Nearly all of the lecture material is helpful in the project. (i.e., >80% of the lecture material)	33%
B	Most of the lecture material is helpful for the project (i.e., > 60% of the lecture material)	0%
C	Much of the lecture material is helpful for the project (i.e., >40% of the lecture material)	33%
D	Some of the lecture material is helpful for the project (i.e., >20% of the lecture material)	0%
E	Hardly any of the lecture material is helpful in the project (i.e., <=20% of the lecture material)	33%

→ `socrative.com`, Room REQENG, Q2 and Q3

Sync of lectures and project, 2023

A	The lecture provide knowledge **much too early** for the project (i.e., the project lags >2 weeks behind the lectures)	0%
B	The lectures provide knowledge **too early** for the project (i.e., the project lags >1 week behind the lectures)	0%
C	The lectures provide knowledge at ***roughly* the right moment** in the project (i.e., +/- 1 week)	86%
D	The lectures provide the knowledge **too late** for the project (i.e., the lectures lag >1 week behind the project)	14%
E	The lectures provide the knowledge **much too late** for the project (i.e., the lectures lag >2 week behind the project)	0%

A	Nearly all of the lecture material is helpful in the project. (i.e., >80% of the lecture material)	14%
B	Most of the lecture material is helpful for the project (i.e., > 60% of the lecture material)	71%
C	Much of the lecture material is helpful for the project (i.e., >40% of the lecture material)	14%
D	Some of the lecture material is helpful for the project (i.e., >20% of the lecture material)	0%
E	Hardly any of the lecture material is helpful in the project (i.e., <=20% of the lecture material)	0%

→ `socrative.com`, Room REQENG, Q2 and Q3

Learning objectives

What are key concepts and characteristics of traceability?

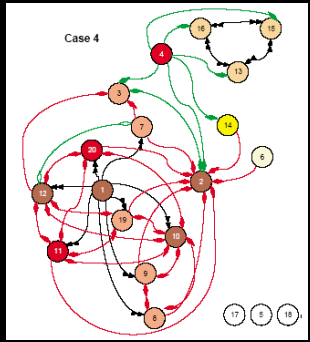
What principles and practices of traceability exist?

Which activities does traceability support?

Why is traeability challenging to achieve?

Outline

[INTDEP]



An industrial survey of requirements interdependencies in software product release planning

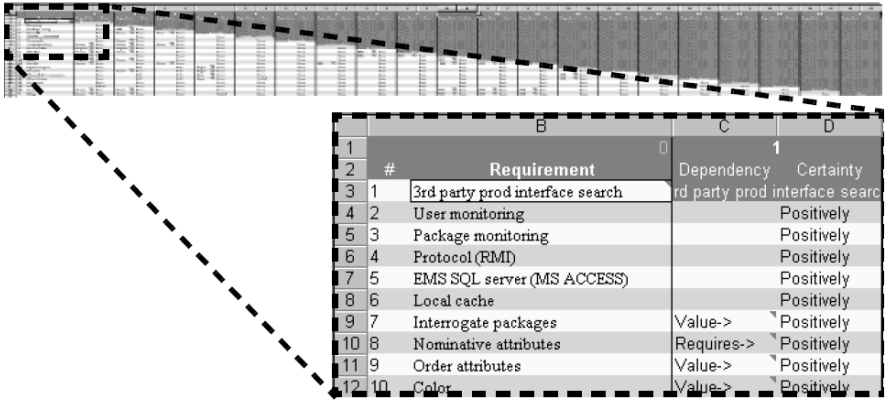
Carlshamre, P., Sandahl, K., Lindvall, M., Regnell, B., Natt och Dag, J.

IEEE Int. Conf. on Requirements Engineering (RE01), Toronto, Canada, pp. 84–91 (2001)

Research Approach

- survey of five different companies
- a manager of a product/project
  was asked to identify and classify interdependencies among
  20 high priority requirements.

Data collection



	B	C	D
1	0		1
2	# Requirement	Dependency	Certainty
3	1 3rd party prod interface search	rd party prod interface searc	
4	2 User monitoring		Positively
5	3 Package monitoring		Positively
6	4 Protocol (RMI)		Positively
7	5 EMS SQL server (MS ACCESS)		Positively
8	6 Local cache		Positively
9	7 Interrogate packages	Value->	Positively
10	8 Nominative attributes	Requires->	Positively
11	9 Order attributes	Value->	Positively
12	10 Color	Value->	Positively

Figure 1. The spreadsheet designed for pairwise assessment of 20 requirements.

Examples of different types of interdependencies

Example 1: AND. A printer requires a driver to function, and the driver requires a printer to function.

Example 2: REQUIRES. Sending an e-mail requires a network connection, but not the opposite.

Example 3: TEMPORAL. The function *Add object* should be implemented before *Delete object*. (This type is doubtful, which is discussed in section 3.1)

Example 4: CVALUE. A detailed on-line manual may decrease the customer value of a printed manual.

Example 5: ICOST. A requirement stating that "no response time should be longer than 1 second" will typically increase the cost of implementing many other requirements.

Example 6: OR. In a word processor, the capability to create pictures in a document can either be provided as an integrated drawing module or by means of a link to an external drawing application.

Not always straight forward ...

- "*if R2 is completely worthless to the customer without R1, and we would thus never do R2 without R1, do we classify the relationship as REQUIRED or just CVALUE?*"
- REQUIRES sometimes arises from the opposite reasoning: "If we do R2, then we can do R1 too!", which implies that the direction of the relationship should be the opposite; could e.g. be called "ENABLES" to signal opposite direction

Summary of identified interdependencies

Table 2. Summary of identified interdependencies.

	# dependencies	most common type	# singular req's	10% of the req's are responsible for	20% of the req's are responsible for	coupling (cf. section 3.5)
Case 1 (prod.)	19	ICOST 79%	4	47% of distinct interdep's	79% of distinct interdep's	10%
Case 2 (prod.)	29	CVALUE 45%	3	55% of distinct interdep's	76% of distinct interdep's	15%
Case 3 (prod.)	42	ICOST 86%	3	50% of distinct interdep's	74% of distinct interdep's	22%
Case 4 (besp.)	41	AND 41%	3	44% of distinct interdep's	71% of distinct interdep's	22%
Case 5 (besp.)	24	REQUIRES 79%	4	42% of distinct interdep's	67% of distinct interdep's	13%

1. 10% of the requirements are responsible for roughly 50% of the interdependencies
2. 20% of the requirements are responsible for roughly 75% of all interdependencies
3. About 20% of the requirements are singular
4. Customer-specific: more functionality-related ; Market-driven: more value-related dependencies

Example of dependency structures

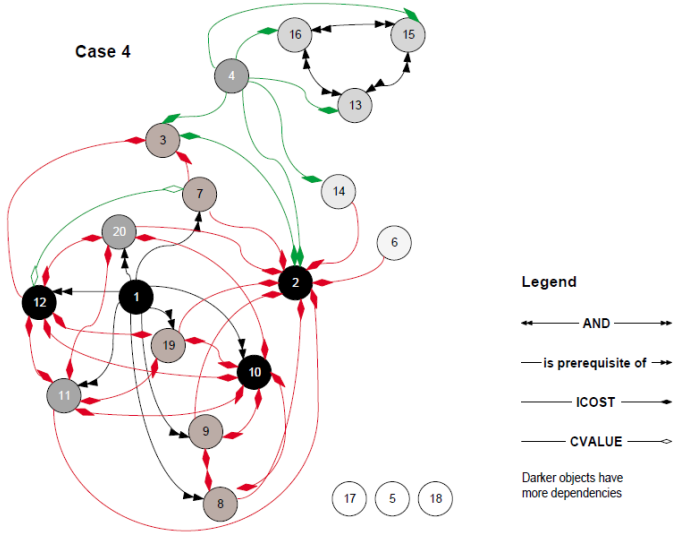


Figure 2. Visualization of requirements interdependecies for one of the five cases.

Coupling measures

$$Creq = \frac{I}{(R(R-1))/2}$$

I =#dependencies
R =#requirements

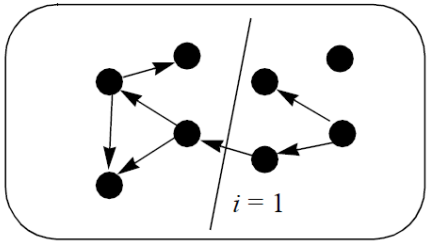
In survey: 10-22%

Release coupling:

$$Crel = \frac{i}{I}$$

i = #dep. betw. 2 partitions

Figure 3. Example illustrating the concepts of requirements and release coupling.



$R = 8$
$I = 7$

$$Creq = \frac{7}{28}$$

$$Crel = \frac{1}{7}$$

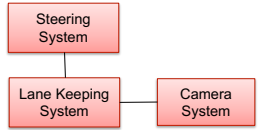
System Requirements

ID	Description
SR 1.1	The **system** shall be able to detect white road lines
SR 1.2	The **system** shall send a signal to the driver when the car is deviating from…
SR 1.3	The **system** shall apply force to the steering wheel to keep the car from …

System Requirements

ID	Description
SR 1.1	The **system** shall be able to detect white road lines
SR 1.2	The **system** shall send a signal to the driver when the car is deviating from…
SR 1.3	The **system** shall apply force to the steering wheel to keep the car from …

System Model



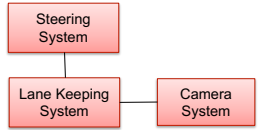
System Requirements

ID	Description
SR 1.1	The **system** shall be able to detect white road lines
SR 1.2	The **system** shall send a signal to the driver when the car is deviating from…
SR 1.3	The **system** shall apply force to the steering wheel to keep the car from …

Software Requirements

ID	Description
SWR 1.1	The **software** should be able to tell road lines from other lines
SWR 1.2	On deviation, the **software** should send a signal to the steering system
SWR 1.3	The **software** should calculate the amount of force to be applied ….

System Model



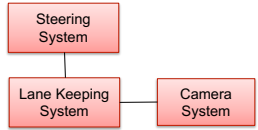
System Requirements

ID	Description
SR 1.1	The **system** shall be able to detect white road lines
SR 1.2	The **system** shall send a signal to the driver when the car is deviating from…
SR 1.3	The **system** shall apply force to the steering wheel to keep the car from …

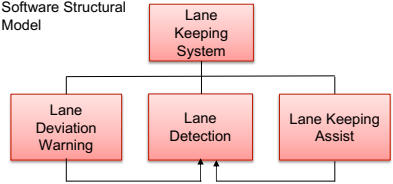
Software Requirements

ID	Description
SWR 1.1	The **software** should be able to tell road lines from other lines
SWR 1.2	On deviation, the **software** should send a signal to the steering system
SWR 1.3	The **software** should calculate the amount of force to be applied ….

System Model



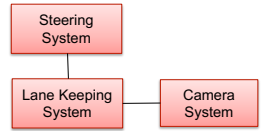
Software Structural Model



System Requirements

ID	Description
SR 1.1	The **system** shall be able to detect white road lines
SR 1.2	The **system** shall send a signal to the driver when the car is deviating from…
SR 1.3	The **system** shall apply force to the steering wheel to keep the car from …

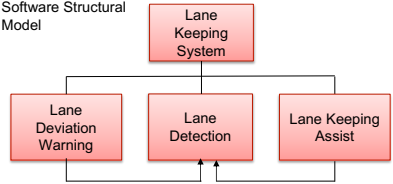
System Model



Software Requirements

ID	Description
SWR 1.1	The **software** should be able to tell road lines from other lines
SWR 1.2	On deviation, the **software** should send a signal to the steering system
SWR 1.3	The **software** should calculate the amount of force to be applied ….

Software Structural Model



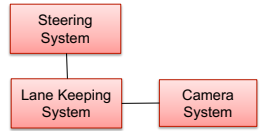
Software Implementation

```
// Name        : LaneKeepingSystem.cpp

using namespace std;

void AlertDriver(double LanePosition, double carPosition) {
    char signal;
    if (LaneDeviated()) {
        signal = "Lane deviated, warning driver";
    }
}
```

System Requirements

ID	Description
SR 1.1	The **system** shall be able to detect white road lines
SR 1.2	The **system** shall send a signal to the driver when the car is deviating from…
SR 1.3	The **system** shall apply force to the steering wheel to keep the car from …

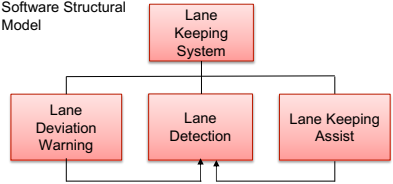
System Model



Software Requirements

ID	Description
SWR 1.1	The **software** should be able to tell road lines from other lines
SR 1.2	On deviation, the **software** should send a signal to the steering system
SR 1.3	The **software** should calculate the amount of force to be applied ….

Software Structural Model

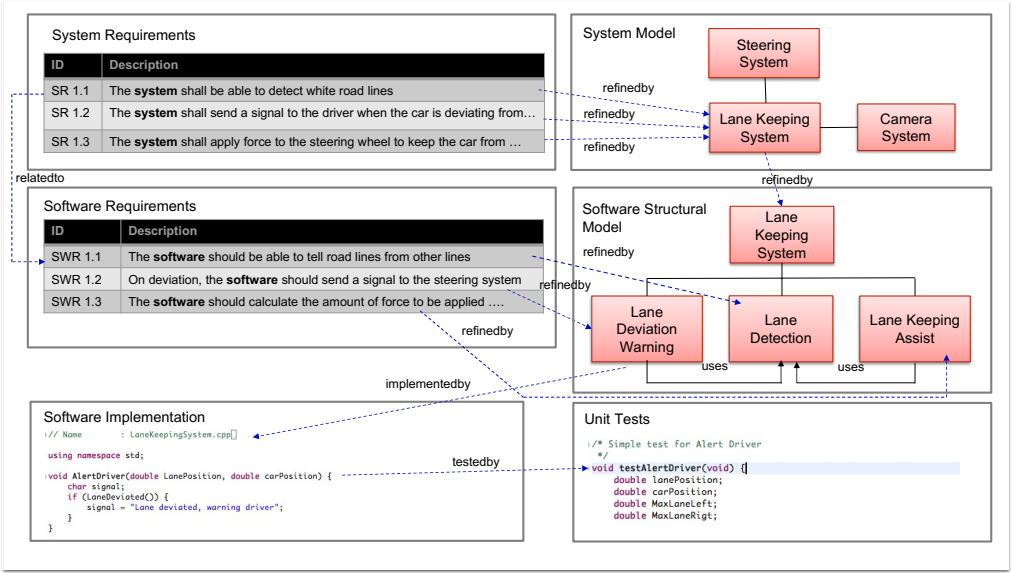


Software Implementation

```
// Name        : LaneKeepingSystem.cpp

using namespace std;

void AlertDriver(double LanePosition, double carPosition) {
    char signal;
    if (LaneDeviated()) {
        signal = "Lane deviated, warning driver";
    }
}
```

Unit Tests

```
/* Simple test for Alert Driver
 */
void testAlertDriver(void) {
    double lanePosition;
    double carPosition;
    double MaxLaneLeft;
    double MaxLaneRigt;
```


System Requirements
ID
Description
SR 1.1
The system shall be able to detect white road lines
SR 1.2
The system shall send a signal to the driver when the car is deviating from…
SR 1.3
The system shall apply force to the steering wheel to keep the car from …
relatedto
Software Requirements
ID
Description
SWR 1.1
The software should be able to tell road lines from other lines
SWR 1.2
On deviation, the software should send a signal to the steering system
SWR 1.3
The software should calculate the amount of force to be applied ….
refinedby
implementedby
Software Implementation
// Name : LaneKeepingSystem.cpp
using namespace std;
void AlertDriver(double LanePosition, double carPosition) {
char signal;
if (LaneDeviated()) {
signal = "Lane deviated, warning driver";
}
}
testedby
System Model
Steering System
Lane Keeping System
Camera System
refinedby
refinedby
refinedby
refinedby
Software Structural Model
Lane Keeping System
refinedby
refinedby
Lane Deviation Warning
Lane Detection
Lane Keeping Assist
uses
uses
Unit Tests
/* Simple test for Alert Driver
*/
void testAlertDriver(void) {
double lanePosition;
double carPosition;
double MaxLaneLeft;
double MaxLaneRigt;

Outline

UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY

Traceability Definition

Definition: Traceability

The ability to relate artifacts created during the development of a system
[Spanoudakis and Zisman, 2005].

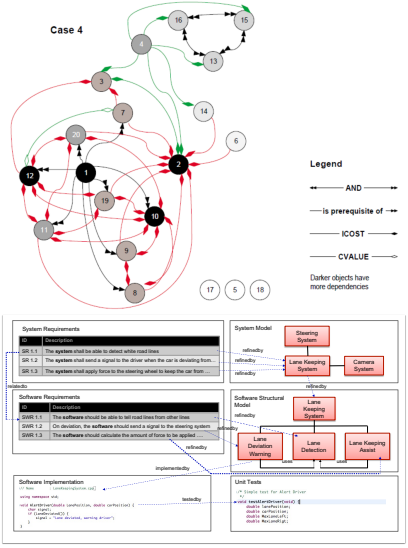
Definition: Horizontal Traceability

Tracing between artifacts of the same type (e.g., requirements) [Ramesh and Edwards, 1993].

Definition: Vertical Traceability

Tracing between artifacts of different types (e.g., requirements and tests)
[Ramesh and Edwards, 1993].

→ `socrative.com`, Room REQENG, Q4

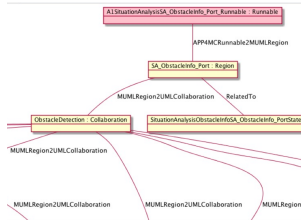


(Some) Purposes of Traceability

Req No	Req Desc	Testcase ID	Status
123	Login to the application	TC01,TC02,TC03	TC01-Pass TC02-Pass
345	Ticket Creation	TC04,TC05,TC06, TC07,TC08,TC09 TC010	TC04-Pass TC05-Pass TC06-Pass TC06-Fail TC07-No Run
456	Search Ticket	TC011,TC012, TC013,TC014	TC011-Pass TC012-Fail TC013-Pass TC014-No Run

https://www.guru99.com/traceability-matrix.html

Test Coverage



Change Impact Analysis

Improved effort estimations

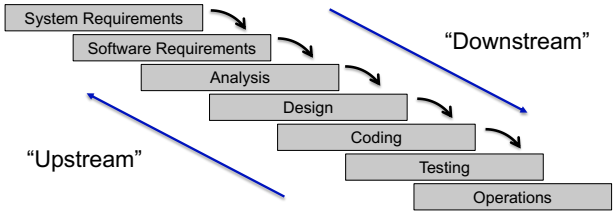
Other Purposes:
- Verification (requirements consistent?),
- Validation (requirements follow from real need?)

Terminology

Definition: Artifact

any tangible work product created in a software development process, i.e., "a traceable unit of data" [Gotel et al., 2012].

Downstream and Upstream Tracing:



→ `socrative.com`, Room REQENG, Q5

Traceability Matrix

		Bug		New Feature			Sub-task				Technical task		Test	
		MAT-1	MAT-9	MAT-10	MAT-13	MAT-14	MAT-2	MAT-3	MAT-4	MAT-6	MAT-5	MAT-7	MAT-8	MAT-12
Bug	MAT-1		✓	✓	✓		✓	✓	✓	✓	✓	✓		
	MAT-9	✓					✓						✓	
New Feature	MAT-10	✓								✓		✓		
	MAT-13	✓				✓								
	MAT-14				✓				✓					
Sub-task	MAT-2	✓	✓											
	MAT-3	✓								✓				
	MAT-4	✓				✓								
	MAT-6	✓		✓				✓					✓	
Technical task	MAT-5	✓												
	MAT-7	✓		✓										
Test	MAT-8		✓							✓				
	MAT-12													

https://marketplace.atlassian.com/apps/1211580/traceability-matrix-and-link-graph?hosting=server&tab=overview

Requirements Traceability

- Traceability from/to requirements
- **Post-requirements traceability:** traceability from requirements, to subsequent artifacts that are derived from the requirements
- **Pre-requirements traceability:** traceability to the source/origin of the requirements. E.g., traceability from requirements to business goals.

Outline

How to use traceability in Requirements Engineering?

- Finding origin and rationale of requirements (pre-requirements traceability)
- Refinement and detailing requirements
- Documenting a requirement's history, i.e., to be able to trace to previous versions of a requirement in order to find out about changes
- Identifying stakeholders for the ongoing development of the requirements
- Impact analysis

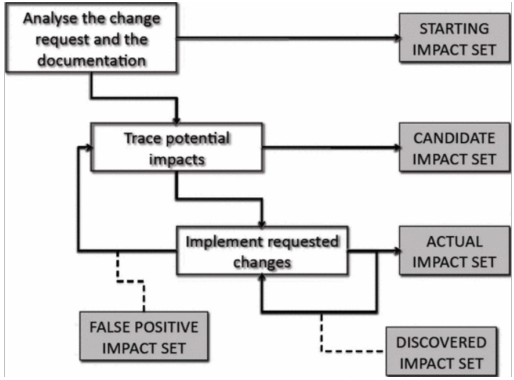
[Bouillon et al., 2013]

Outline

Change Impact Analysis

- Changes are triggered by new or altered requirements and usually captured in a change request
- Changes have impact on all artefacts, not only source code
- Traceability allows to locate all artefacts that are impacted by a change
- Goal: candidate impact set = actual impact set

[De Lucia et al., 2008]



[Bohner, 2002]

GDT Questions [Lauesen, 2002, pg.364]

1. For each business goal: which tasks / quality factors ensure that goal can be met?
2. For each task and quality factor: what is its purpose in terms of business goals?

→ `socrative.com`, Room REQENG, Q5

Goal Domain Tracing

Fig 8.7A Goal-domain tracing

	Marketing	Quotation	Production planning	Cost registration	Invoicing	. . .	Payroll	IT operations	Usability	Response time
Replace IT platform							●	●		
Integrate doc and data	●	●		●						
Experience data		●		●					●	●
Systematic marketing	●	●								
Capture cost data				●	●		●	●		
Speed up invoicing		●			●				●	●

Above: Exampel from [Lauesen, 2002, pg.365]
Below: Example from our SensIT example project

Goal	(e1) view	(e2) find hotspots	(e3) annotate	(e4) overlay	(qr1) accuracy
Business Goal 1.	x	x	x	x	x
Better inspection.					
Business Goal 2.			x	x	x
Better inspection report.					

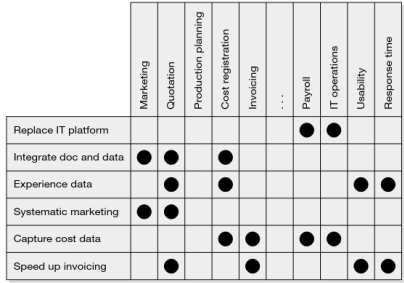
GDT Questions [Lauesen, 2002, pg.364]

1. For each business goal: which tasks / quality factors ensure that goal can be met?
2. For each task and quality factor: what is its purpose in terms of business goals?

Why to do GDT?

Goal Domain Tracing

Fig 8.7A Goal-domain tracing



Above: Exampel from [Lauesen, 2002, pg.365]
Below: Example from our SensIT example project

Goal	(e1) view	(e2) find hotspots	(e3) annotate	(e4) overlay	(qr1) accuracy
Business Goal 1.	x	x	x	x	x
Better inspection.					
Business Goal 2.			x	x	x
Better inspection report.					

GDT Questions [Lauesen, 2002, pg.364]

1. For each business goal: which tasks / quality factors ensure that goal can be met?
2. For each task and quality factor: what is its purpose in terms of business goals?

Why to do GDT?

- Elicitation: Guidance, e.g. understand quality factor / task in context of business goals
- Validation: Tasks and quality factors relate to real need?
- Verification: Documentation complete and consistent?

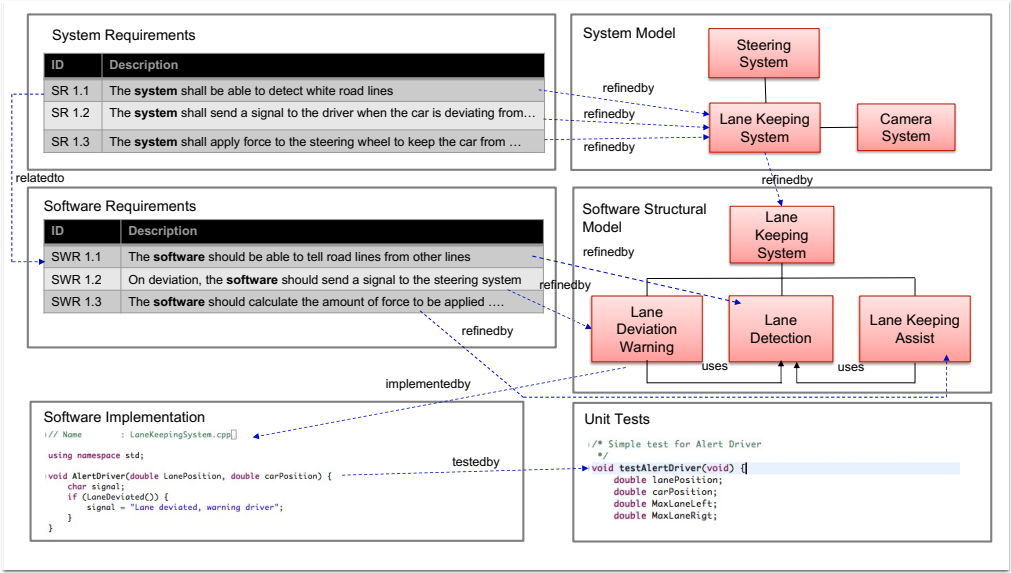
Goal Domain Tracing

Fig 8.7A Goal-domain tracing

	Marketing	Quotation	Production planning	Cost registration	Invoicing	. . .	Payroll	IT operations	Usability	Response time
Replace IT platform							●	●		
Integrate doc and data	●	●		●						
Experience data		●		●					●	●
Systematic marketing	●	●								
Capture cost data				●	●		●	●		
Speed up invoicing		●			●				●	●

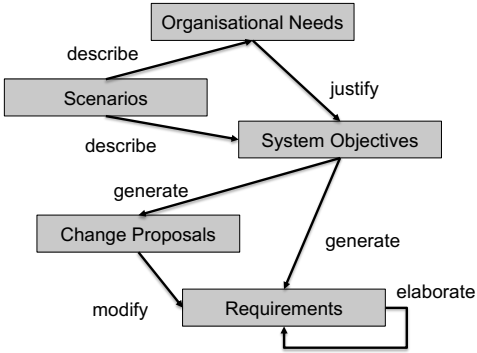
Above: Exampel from [Lauesen, 2002, pg.365]
Below: Example from our SensIT example project

Goal	(e1) view	(e2) find hotspots	(e3) annotate	(e4) overlay	(qr1) accuracy
Business Goal 1.	x	x	x	x	x
Better inspection.					
Business Goal 2.			x	x	x
Better inspection report.					

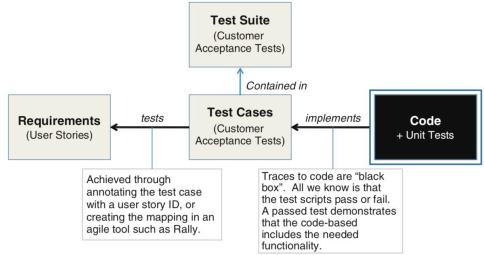

System Requirements
ID
Description
SR 1.1
The system shall be able to detect white road lines
SR 1.2
The system shall send a signal to the driver when the car is deviating from…
SR 1.3
The system shall apply force to the steering wheel to keep the car from …
System Model
Steering System
Lane Keeping System
Camera System
refinedby
refinedby
refinedby
relatedto
refinedby
Software Requirements
ID
Description
SWR 1.1
The software should be able to tell road lines from other lines
SWR 1.2
On deviation, the software should send a signal to the steering system
SWR 1.3
The software should calculate the amount of force to be applied ….
refinedby
Software Structural Model
Lane Keeping System
refinedby
refinedby
Lane Deviation Warning
Lane Detection
Lane Keeping Assist
uses
uses
implementedby
Software Implementation
// Name : LaneKeepingSystem.cpp
using namespace std;
void AlertDriver(double LanePosition, double carPosition) {
char signal;
if (LaneDeviated()) {
signal = "Lane deviated, warning driver";
}
}
testedby
Unit Tests
/* Simple test for Alert Driver
*/
void testAlertDriver(void) {
double lanePosition;
double carPosition;
double MaxLaneLeft;
double MaxLaneRigt;

Traceability Information Models

Describe the semantics of tracelinks



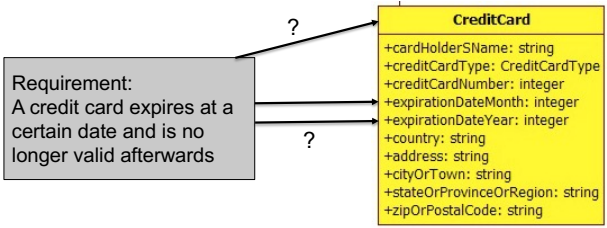
[Ramesh and Jarke, 2001]



[Cleland-Huang, 2012]

Traceability Information Models

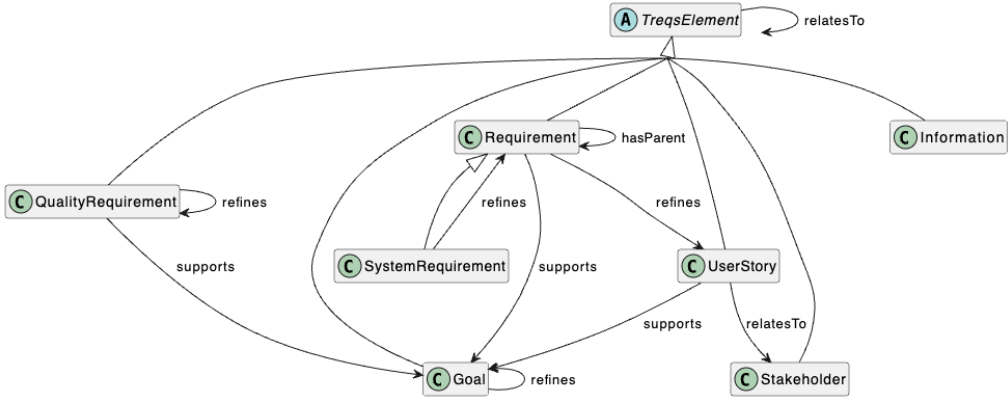
Which level of granularity?



Level of granularity depends on desired semantics and has an impact on generality and maintainability.

https://upload.wikimedia.org/wikipedia/commons/c/c6/Facebook_Metamodel_Represented_in_UML_notation_v01.jpg

Traceability Information Model for Course Template

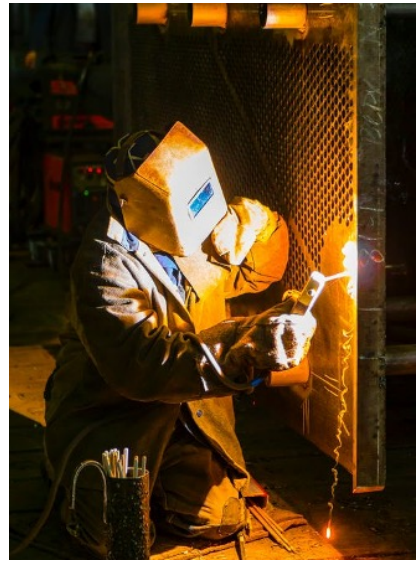


Outline

Traceability Maintenance

- Artifacts evolve in a development process – trace links need to evolve with them
- Stale links make all traceability activities difficult
- Relevant changes to artifacts need to be recognised
- Trace links need to be updated in response
- Recording and validating these updates is tedious and time-consuming

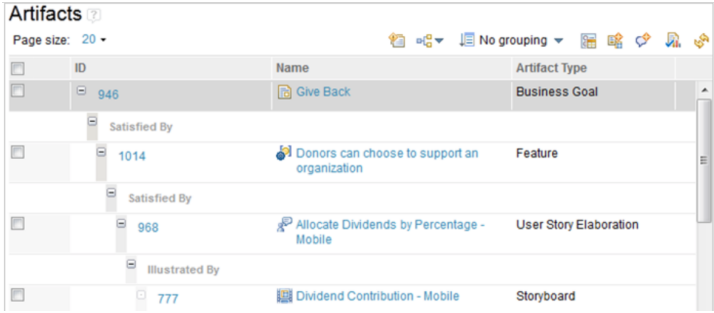
"I'd rather have no links than one wrong link"
—Engineer from the automotive industry



Traceability Tools

- Manual tracing using rudimentary tools is tedious and therefore specialized tools are needed
- Many requirements management tools offer traceability support e.g. IBM DOORS
- Common features of traceability tools:
  - Traceability visualization e.g., matric view, graphical views
  - Suspect link detection (to facilitate maintenance)
  - Traceability reports (e.g., which artifacts have missing links)

Traceability Tools



Example of a traceability visualization in IBM Jazz

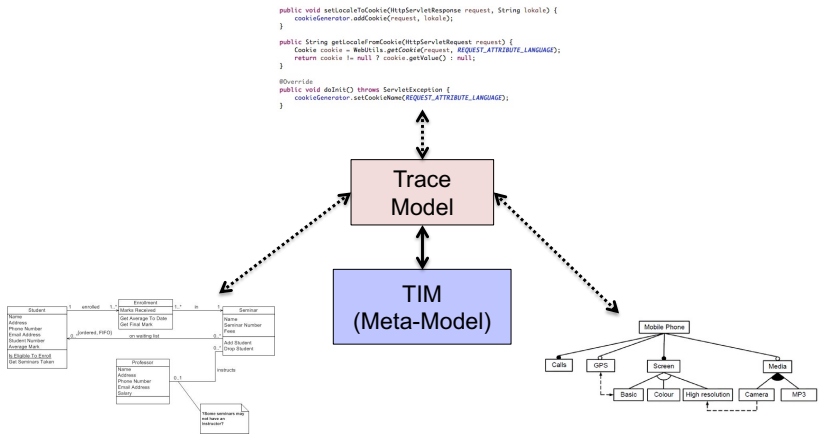
https://jazz.net/library/article/88104

Outline

State of the Art

- Model-driven Traceability [Santiago et al., 2013]
- Active local research
  - CAPRA (Eclipse based traceability tool) via Jan-Philipp Steghöfer
  - TReqs (CLI and git based traceability management) via Eric Knauss
  - Collaborative Traceability (traceability across boundaries for mutual benefit) [Wohlrab et al., 2020]
  - Traceability Information Models for Agile [Maro et al., 2020]

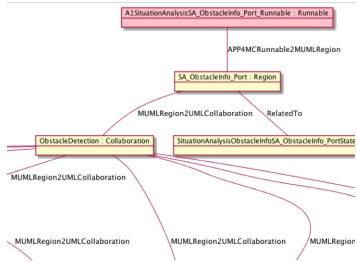
Model-Driven Traceability



[Santiago et al., 2013]

Model-Driven Traceability

Req No	Req Desc	Testcase ID	Status
123	Login to the application	TC01,TC02,TC03	TC01-Pass TC02-Pass
345	Ticket Creation	TC04,TC05,TC06, TC07,TC08,TC09 TC010	TC04-Pass TC05-Pass TC06-Pass TC06-Fail TC07-No Run
456	Search Ticket	TC011,TC012, TC013,TC014	TC011-Pass TC012-Fail TC013-Pass TC014-No Run



[Santiago et al., 2013]

Outline

Learning objectives

What are key concepts and characteristics of traceability?

- The ability to relate artifacts during development of a system (horizontal/vertical traceability; artifact; pre-/post-requirements traceability

What principles and practices of traceability exist?

Which activities does traceability support?

Why is traeability challenging to achieve?

Learning objectives

What are key concepts and characteristics of traceability?

- The ability to relate artifacts during development of a system (horizontal/vertical traceability; artifact; pre-/post-requirements traceability

What principles and practices of traceability exist?

- Use Goal-Domain Tracing, Traceability Matrices, Traceability Information Models, Query Languages
- Rely on a Traceability Information Model to define which tracelinks can or must exist
- Use tools to maintain tracelinks

Which activities does traceability support?

Why is traeability challenging to achieve?

Learning objectives

What are key concepts and characteristics of traceability?

- The ability to relate artifacts during development of a system (horizontal/vertical traceability; artifact; pre-/post-requirements traceability

What principles and practices of traceability exist?

- Use Goal-Domain Tracing, Traceability Matrices, Traceability Information Models, Query Languages
- Rely on a Traceability Information Model to define which tracelinks can or must exist
- Use tools to maintain tracelinks

Which activities does traceability support?

- Apply traceability driven by a need (Change Impact Analysis, Requirements Elicitation or Requirements V&V, Coverage Analysis, ...)
- Often needed to prove compliance to a standard or regulation

Why is traeability challenging to achieve?

Learning objectives

What are key concepts and characteristics of traceability?

- The ability to relate artifacts during development of a system (horizontal/vertical traceability; artifact; pre-/post-requirements traceability

What principles and practices of traceability exist?

- Use Goal-Domain Tracing, Traceability Matrices, Traceability Information Models, Query Languages
- Rely on a Traceability Information Model to define which tracelinks can or must exist
- Use tools to maintain tracelinks

Which activities does traceability support?

- Apply traceability driven by a need (Change Impact Analysis, Requirements Elicitation or Requirements V&V, Coverage Analysis, ...)
- Often needed to prove compliance to a standard or regulation

Why is traeability challenging to achieve?

- Bad tracelinks worse than no tracelinks,
- ...expensive to create and maintain,
- Tracing often an after thought

Summary

- Traceability makes your life easier!
- Traceability is purpose-driven
- Trace links need to have clear semantics aligned with the purpose
- Creating and maintaining trace links is difficult and requires continuous effort
- Model-driven traceability allows deriving different views on trace links for different purposes

Attribution:
– These slides are adopted from Jan-Philipp Steghöfer and Salome Maro
– all pictures by pexels.com if not otherwise noted

References I


Bohner, S. A. (2002).
Software change impacts: An evolving perspective.
In *Proc. 18th ICSM*, pages 263–272.


Bouillon, E., Mäder, P., and Philippow, I. (2013).
A survey on usage scenarios for requirements traceability in practice.
In *Proc. of International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, Berlin, Heidelberg.


Cleland-Huang, J. (2012).
Software and Systems Traceability, chapter Traceability in agile projects, pages 265–275.
Springer, London.


De Lucia, A., Fasano, F., and Oliveto, R. (2008).
Traceability management for impact analysis.
Frontiers of Software Maintenance.


Gotel, O., Cleland-Huang, J., Hayes, J. H., Zisman, A., Egyed, A., Grünbacher, P., Dekhtyar, A., Antoniol, G., Maletic, J., and Mäder, P. (2012).
Software and Systems Traceability, chapter 1: Traceability fundamentals, pages 3–22.
Springer, London.


Lauesen, S. (2002).
Software Requirements.
Pearson / Addison-Wesley.
the course book (Lau).

References II

Maro, S. H., Steghöfer, J.-P., Knauss, E., Horkoff, J., Kasauli, R., Korsgaard, J. L., Wartenberg, F., Strøm, N. J., and Alexandersson, R. (2020).
Managing traceability information models: Not such a simple task after all.
IEEE Software, 38(5):101–109.

Ramesh, B. and Edwards, M. (1993).
Issues in the development of a requirements traceability model.
In *Proceedings of the IEEE International Symposium on Requirements Engineering*. IEEE.

Ramesh, B. and Jarke, M. (2001).
Toward reference models for requirements traceability.
Transactions on Software Engineering, 27(1):58–93.

Santiago, I., Vara, J. M., de Castro, V., and Marcos, E. (2013).
Towards the effective use of traceability in model-driven engineering projects.
In *Proc. of International Conference on Conceptual Modeling*. Springer, Berlin, Heidelberg.

Spanoudakis, G. and Zisman, A. (2005).
Handbook of Software Engineering and Knowledge Engineering, chapter 3: Software traceability: a roadmap, pages 395–428.

Wohlrab, R., Knauss, E., Steghöfer, J.-P., Maro, S., Anjorin, A., and Pelliccione, P. (2020).
Collaborative traceability management: A multiple case study from the perspectives of organization, process, and culture.
Requirements Engineering (REEN), 25:21–45.