



Automatic Validation and Correction of Formalized, Textual Requirements

Jörg Holtmann, Jan Meyer, Markus von Detten
ReVVerT 2011, March 25, 2011



HEINZ NIXDORF INSTITUT
Universität Paderborn
Softwaretechnik
Prof. Dr. Wilhelm Schäfer

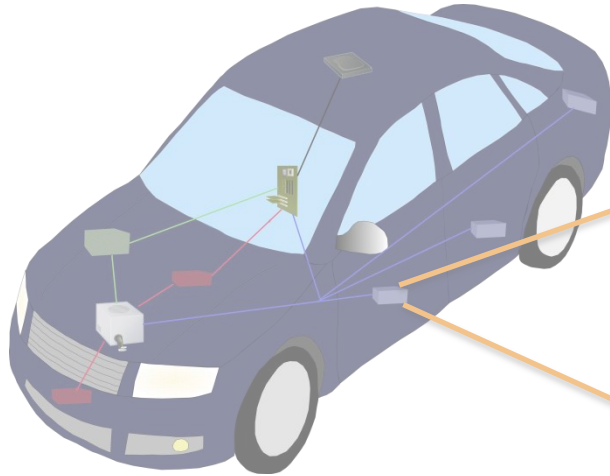


Software Plattform Embedded Systems 2020

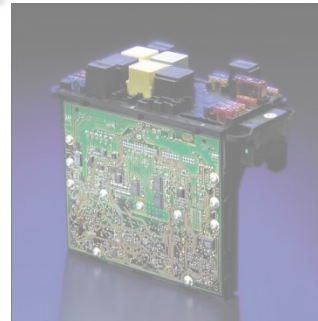
Introduction



Original Equipment
Manufacturer (OEM)



Supplier



Comfort Control Unit



Central
Locking



Indicator
Lights



Interior
Light Control

OEM



1. Customer requirements



2. System requirements



Supplier



System requirements specification



4 Functionality

4.1 System Comfort Control Unit



Manual analysis for rule violations: ☹

4.1.1 System Central Locking System

4.1.1.1 System Door Unlocking

4.1.1.1.1 Function Unlock Door L

4.1.1.1.2 Function Unlock Door R

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 Function Safety Lock L

4.1.1.2.1.2 Function Safety Lock R

4.1.1.2.2 Function Lock Door L

4.1.1.2.3 Function Lock Door R

4.1.2 System Interior Light Control

...

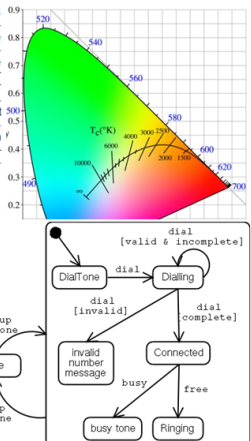
4.1.3 System Window Lifter

...

- Inputs
- Outputs
- Functionality description

Anforderungen werden heutzutage – zumindest bei den Industriepartnern in SPES 2020 – hauptsächlich natürlichsprachlich beschrieben [DLST10]. Im Bereich Automotive liegt dies u.a. daran, dass das Pflichtenheft als vertragliche Grundlage für alle beteiligten Stakeholder, wie Original Equipment Manufacturers (OEMs) und Zulieferer, dient. Um eine leicht prüfbare, juristisch vertretbare und dokumentorientierte Form zu erreichen und den Lernaufwand für spezifische Modellierungssprachen für alle beteiligten Stakeholder zu minimieren, werden die Anforderungen in natürlicher Sprache verfasst [Hol10], da diese keine Schulungen oder spezielle Werkzeuge erfordert [Poh07] [Bal09]. Der Nachteil natürlicher Sprache ist, dass sie inhärent mehrdeutig und nicht formal ist und somit zum einen von verschiedenen Stakeholdern unterschiedlich verstanden [Poh07] [Bal09] und zum anderen nicht automatisch verarbeitet werden kann [HN10] [Hol10] [YBL10]. Letzteres bedingt manuelle Analysen von Anforderungsspezifikationen wie z.B. Reviews, die aufwändig und fehleranfällig sind.

Regeln:	erfüllt-Regel	Hysterese-Regel	Nicht-erfüllt-Regel	(Sond-Regel)
Betriebsbedingungen:				
"<Betriebsbedingung 1>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 2>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 3>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 4>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 5>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 6>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung n>" == "<Ausprägung>"	?	?	?	?
Ereignisse gemäß den Regeln:	UND-Verknüpfung	UND-Verknüpfung	ODER-Verknüpfung	
"<OCC_Betriebsbedingung_Status>" wechselt auf "erfüllt"	X			
keine Änderung (z.B. wenn ein Betriebsbedingung im Hysteresebereich ist)		X		
"<OCC_Betriebsbedingung_Status>" wechselt auf "nicht-erfüllt"			X	



Rule wrt. **readability** [Poh10] of overall specification:
Functionality description in leaf systems!

➔ System consists either of subsystems or functions! [Kap07]

System requirements specification



4 Functionality

4.1 System Comfort Control Unit



Manual correction of defective requirements: ☹

4.1.1 System Central Locking System

4.1.1.1 System Door Unlocking

4.1.1.1.1 Function Unlock Door L

4.1.1.1.2 Function Unlock Door R

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 Function Safety Lock L

4.1.1.2.1.2 Function Safety Lock R

4.1.1.2.2 System Normal Locking

4.1.1.2.2.1 Function Lock Door L

4.1.1.2.2.3 Function Lock Door R

4.1.2 System Interior Light Control

...

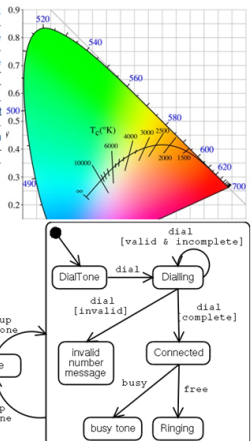
4.1.3 System Window Lifter

...

- Inputs
- Outputs
- Functionality description

Anforderungen werden heutzutage – zumindest bei den Industriepartnern in SPES 2020 – hauptsächlich natürlichsprachlich beschrieben [DLST10]. Im Bereich Automotive liegt dies u.a. daran, dass das Pflichtenheft als vertragliche Grundlage für alle beteiligten Stakeholder, wie Original Equipment Manufacturers (OEMs) und Zulieferern, dient. Um eine leicht prüfbare, juristisch vertretbare und dokumentorientierte Form zu erreichen und den Lernaufwand für spezifische Modellierungssprachen für alle beteiligten Stakeholder zu minimieren, werden die Anforderungen in natürlicher Sprache verfasst [Hol10], da diese keine Schulungen oder spezielle Werkzeuge erfordert [Poh07] [Bal09]. Der Nachteil natürlicher Sprache ist, dass sie inhärent mehrdeutig und nicht formal ist und somit zum einen von verschiedenen Stakeholdern unterschiedlich verstanden [Poh07] [Bal09] und zum anderen nicht automatisch verarbeitet werden kann [HN10] [Hol10] [YBL10]. Letzteres bedingt manuelle Analysen von Anforderungsspezifikationen wie z.B. Reviews, die aufwändig und fehleranfällig sind.

Regeln:	erfüllt-Regel	Hysterese-Regel	Nicht-erfüllt-Regel	Sonst.-Regel
Betriebsbedingungen:				
"<Betriebsbedingung 1>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 2>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 3>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 4>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 5>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung 6>" == "<Ausprägung>"	?	?	?	?
"<Betriebsbedingung n>" == "<Ausprägung>"	?	?	?	?
Ereignisse gemäß den Regeln:	UND-Verknüpfung	UND-Verknüpfung	ODER-Verknüpfung	
"<OCC_Betriebsbedingung_Status>" wechselt auf "erfüllt"	X			
keine Änderung (z.B. wenn ein Betriebsbedingung im Hysteresebereich ist)		X		
"<OCC_Betriebsbedingung_Status>" wechselt auf "nicht-erfüllt"				X



Rule wrt. **readability** [Poh10] of overall specification:
Functionality description in leaf systems!

➔ System consists either of subsystems or functions! [Kap07]

- Requirements validation inevitable
- Manual requirements validation techniques (e.g., reviews):
 - Time-consuming
 - Error-prone
 - Repetitive
- Applies to manual correction of defective requirements, too

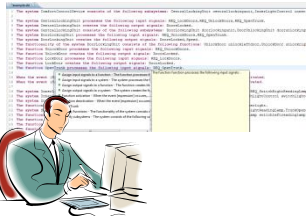


Automated validation and correction of textual requirements needed!



- But: unrestricted natural language (NL) informal and ambiguous
 - ➔ NL requirements cannot be automatically processed
 - ➔ More formal requirements representation required

System
requirements
(CNL)

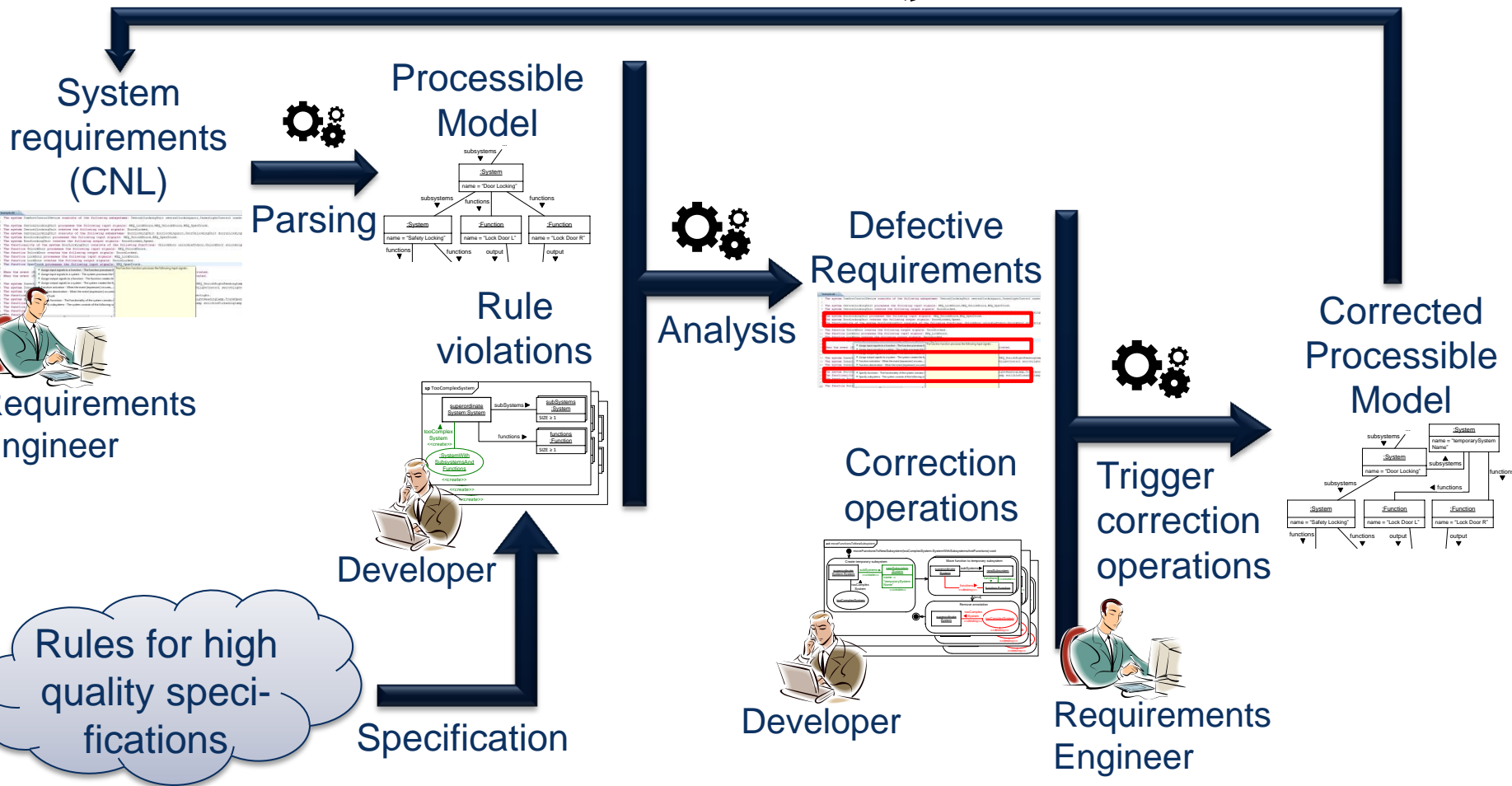


Requirements
Engineer

- Use of Controlled Natural Language (CNL) [KK06] for system requirements specifications
 - No arbitrary prose
 - Requirements editor [Hol10]
 - Error marking
 - Auto completion
 - Syntax Highlighting
 - Enable automatic processing
- ➔ Basis for automated requirements validation

Solution idea (2/2)

Text synthesis 



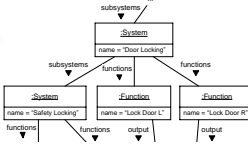
Parsing into processible model (1/2)

Text synthesis 

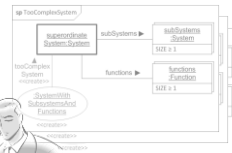
System
requirements
(CNL)


Parsing

Processible
Model



Rule
violations



Developer

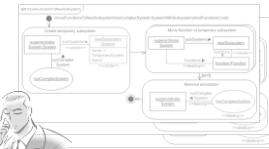
Specification


Analysis


Defective
Requirements



Correction
operations



Developer


Trigger
correction
operations

Corrected
Processible
Model



Requirements
Engineer

Rules for high
quality speci-
fications

Requirements formulated according to CNL are parsed into Abstract Syntax Graph (ASG)

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 ...

4.1.1.2.1.2 ...

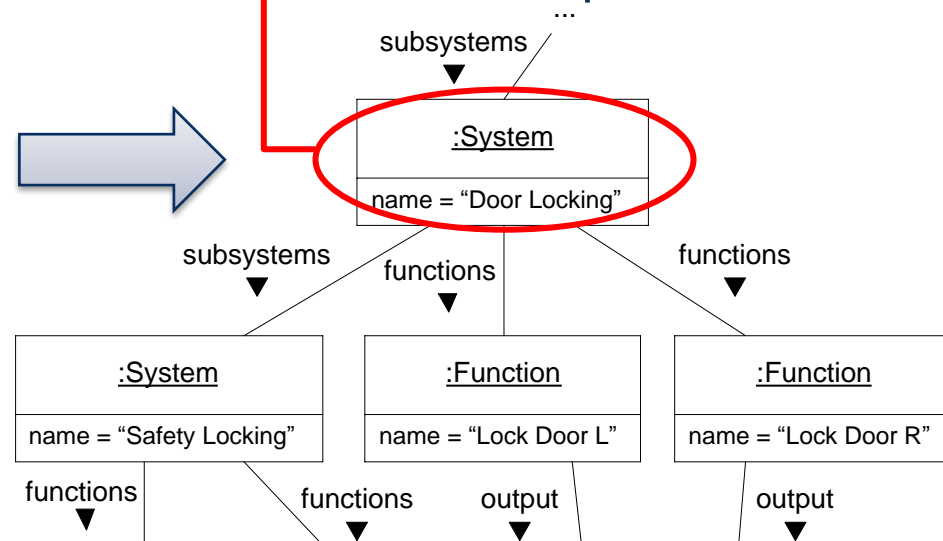
4.1.1.2.2 Function Lock Door L

4.1.1.2.3 Function Lock Door R

Concrete requirements

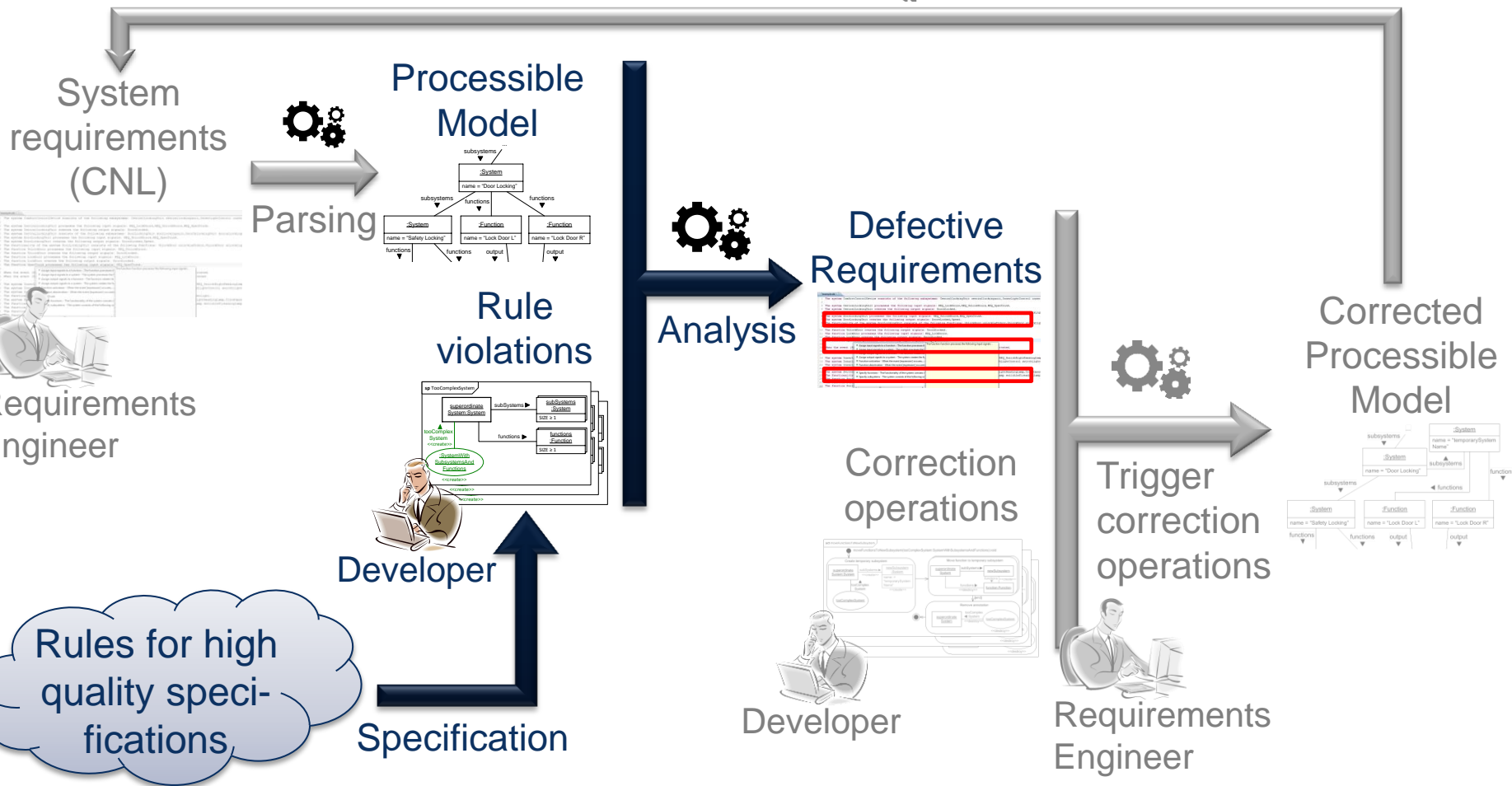
- The system “Door Locking” consists of the following subsystem: Safety Locking.
- The functionality of the system “Door Locking” consists of the following functions: Lock Door L, Lock Door R.

ASG excerpt



Requirements Validation (1/3)

Text synthesis 



- Rule: Functions should only appear in leaf systems
 - Systems can consist of subsystems and functions
 - Functions are “atomic” (do not consist of further elements)
- ➔ Rule satisfied, if systems consist either of subsystems or of functions
- ➔ Rule violation: System consists of subsystems as well as functions

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 ...

4.1.1.2.1.2 ...

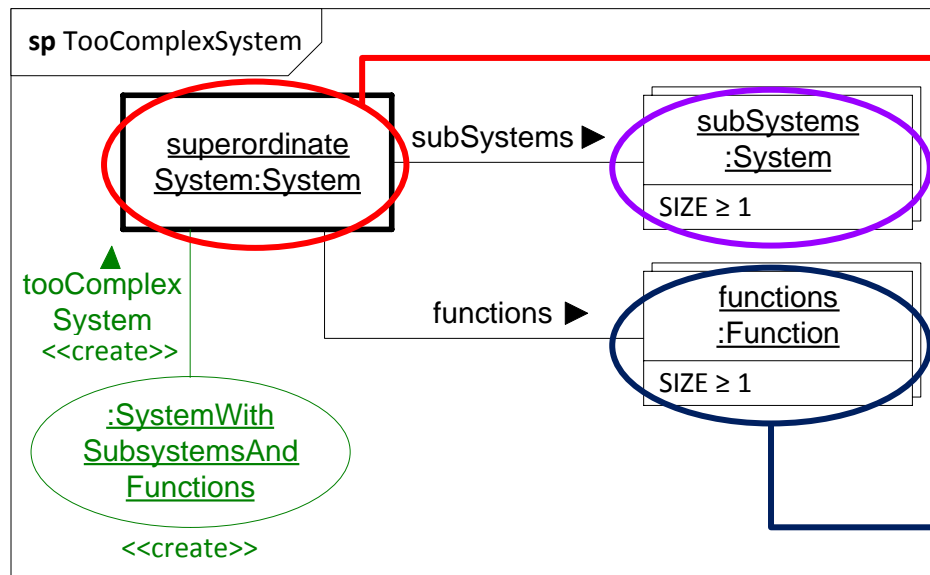
4.1.1.2.2 Function Lock Door L

4.1.1.2.3 Function Lock Door R

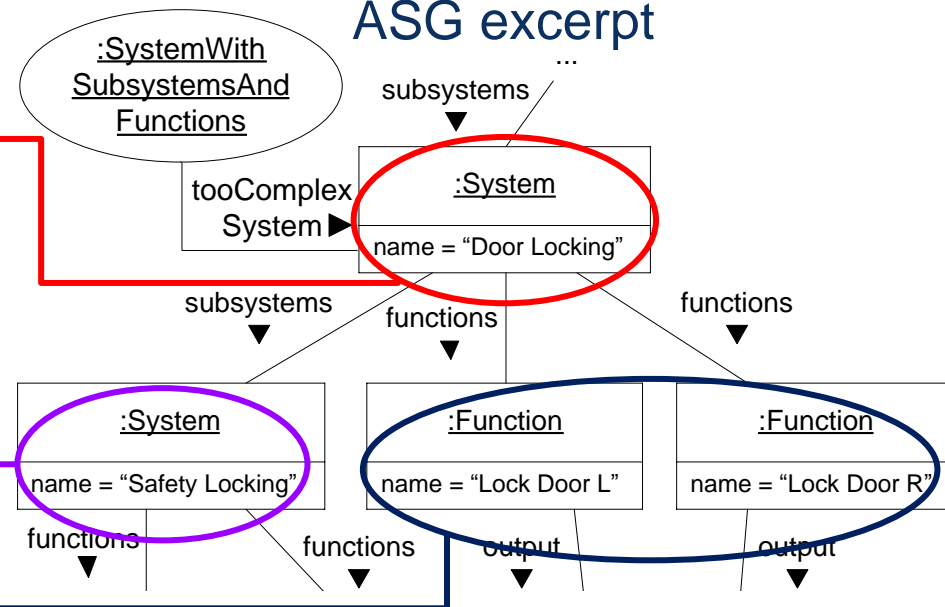


- Graph transformation (GT) pattern for rule violation: system consists of subsystems as well as functions
- If matched: annotation node is created
 - Tagging as defective requirement
 - Information which rule is violated

GT pattern „TooComplexSystem“

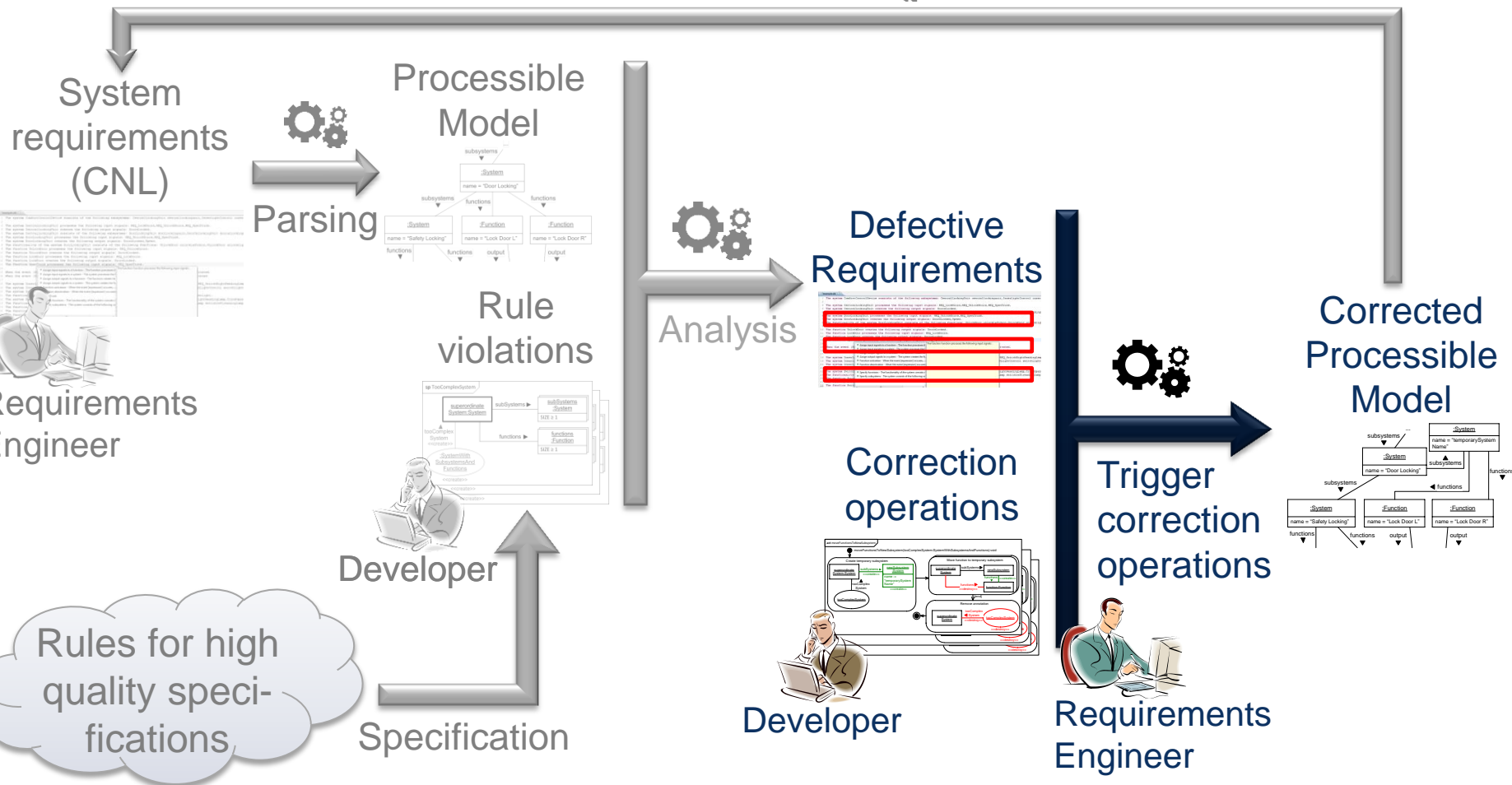


ASG excerpt



Correction operations (1/3)

Text synthesis 



- Strategies to resolve rule violation
 - Move subsystems one level higher
 - **Move functions to new subsystem**

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 ...

4.1.1.2.1.2 ...

4.1.1.2.2 Function Lock Door L

4.1.1.2.3 Function Lock Door R



4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 ...

4.1.1.2.1.2 ...

4.1.1.2.2 System Normal Locking

4.1.1.2.2.1 Function Lock Door L

4.1.1.2.2.2 Function Lock Door R

4 Functionality

4.1 System Comfort Control Unit

4.1.1 System Central Locking System

4.1.1.1 System Door Unlocking

4.1.1.1.1 Function Unlock Door L

4.1.1.1.2 Function Unlock Door R

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 Function Safety Lock L

4.1.1.2.1.2 Function Safety Lock R

4.1.1.2.2 System Normal Locking

4.1.1.2.2.1 Function Lock Door L

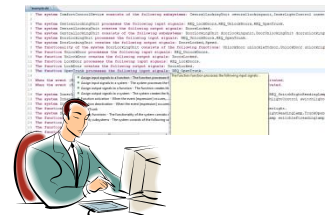
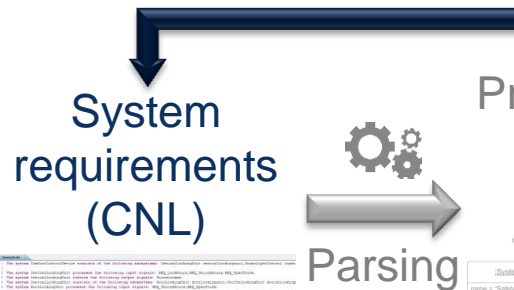
4.1.1.2.2.2 Function Lock Door R

4.1.2 System Interior Light Control

...

4.1.3 System Window Lifter

...



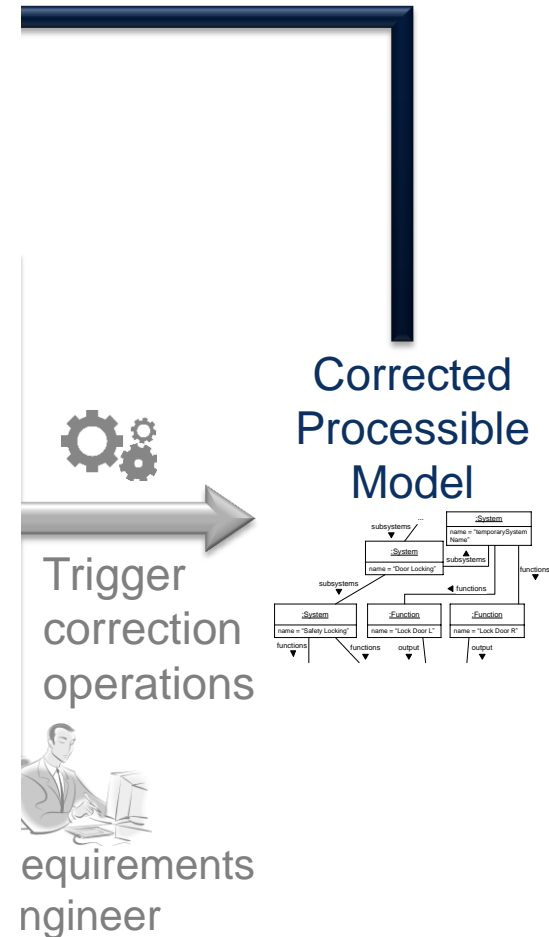
Requirements Engineer



Developer

Rules for high quality specifications

Specifications



- Refer to overall structure of described function hierarchy
- Stem from experiences using the CNL
- Further examples
 - (De-)Activation events/conditions wrt. signals have to be within signal range → consistency
 - Functions/subsystems must only use signals provided by superordinate system → completeness, consistency
 - Empty systems → completeness
 - Functions without inputs/outputs → completeness
 - Specified signals have to be referenced → completeness
 - Not used system signals → completeness
 - Overlapping (de-)activation events for the same function → consistency

4 Functionality

4.1 System Comfort Control Unit

4.1.1 System Central Locking System

4.1.1.1 System Door Unlocking

4.1.1.1.1 Function Unlock Door L

4.1.1.1.2 Function Unlock Door R

4.1.1.2 System Door Locking

4.1.1.2.1 System Safety Locking

4.1.1.2.1.1 Function Safety Lock L

4.1.1.2.1.2 Function Safety Lock R

4.1.1.2.2 System Normal Locking

4.1.1.2.2.1 Function Lock Door L

4.1.1.2.2.2 Function Lock Door R

4.1.2 System Interior Light Control

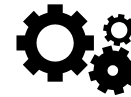
...

4.1.3 System Window Lifter

...

- Limitations
 - No assurance of semantically complete specifications
 - Not every inconsistency can be found
- Benefits
 - Several rules for high quality specifications
 - For each rule: one rule violating pattern
 - If possible: one or several alternative correction operations
 - ➔ Rule violation found: hint that something is wrong
 - ➔ If no rule is violated, the specification is on a good way

- Conclusion
 - Controlled Natural Language (CNL)
 - No arbitrary prose → constructive defect avoidance
 - Enable automatic processing of textual requirements
 - Automated requirements validation
 - Rules for high quality specifications
 - Graph transformation patterns for detection of rule violating requirements
 - Further guidance by automatic correction operations for defective requirements (also based on graph transformations)
- Outlook
 - Tool support and evaluation
 - Transition to model-based design
 - Transfer of the general concept to other domains
 - Transition to testing



- [FNTZ00] T. Fischer, J. Niere, L. Torunski, and A. Zündorf: **Story Diagrams: A new Graph Rewrite Language based on the Unified Modeling Language**. In Theory and Application of Graph Transformations, Lecture Notes in Computer Science (LNCS) 1764, pp. 157–167, 2000
- [Hol10] J. Holtmann: **Mit Satzmustern von textuellen Anforderungen zu Modellen**. In OBJEKTSpektrum, no. RE/2010 (Online Themenspecial Requirements Engineering). SIGS DATACOM, 2010
- [Kap07] R. Kapeller: **Einsatz einer Basisontologie für das Reverse Software Engineering im Rahmen des Anforderungsmanagements für Produktlinien**. In FG SRE: Bericht und Beiträge vom Workshop *Software-Reengineering (WSR 2007)*, 2007
- [KK06] R. Kapeller and S. Krause: **So natürlich wie Sprechen – Embedded Systeme modellieren**. In Design & Elektronik, no. 8, 2006.
- [Poh10] K. Pohl: **Requirements Engineering. Fundamentals, Principles, and Techniques**. Springer, 2010
- [vDMT10] M. von Detten, M. Meyer, and D. Travkin: **Reverse Engineering with the Reclipse Tool Suite**. In Proc. of the 32nd ACM/IEEE International Conference on Software Engineering (ICSE 2010), 2010

Thanks for your attention.

s-lab – Software Quality Lab

University of Paderborn

Warburger Str. 100

33098 Paderborn

Tel.: (05251) 60 5390 / 5391

<http://s-lab.upb.de>

info@s-lab.upb.de

