Machine-Checked Compositional Specification and Proofs for Embedded Systems

Karl Palmskog* Mattias Nyberg† Dilian Gurov*

*KTH Royal Institute of Technology, Sweden

†Scania CV AB, Sweden

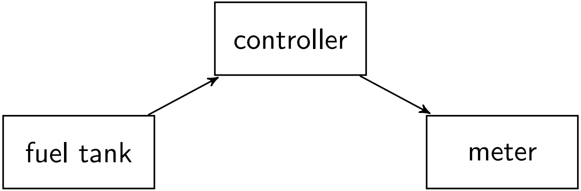
2025-07-14

Components, Specifications and Compositionality

- component c is composed of components $c_1, c_2, \ldots, c_n$
- we want to ensure c implements specification S
- we find specifications $S_1, S_2, \ldots, S_n$ and verify that
  - c_1 implements S_1
  - c_2 implements S_2
  - $\ldots$
  - c_n implements S_n
- is this sufficient for establishing that c implements S?

Embedded Systems and Compositionality

- Scania heavy vehicles are composed of (tens of) ECUs
- ECUs can communicate over wires (e.g., CAN bus)
- collections of ECUs may need to implement a specification
- example: does meter (eventually) show the fuel level?



Previous Work

Formally Proving Compositionality in Industrial Systems with Informal Specifications

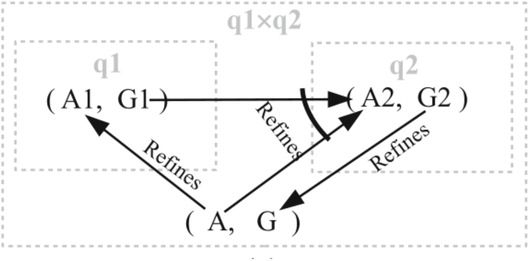
Mattias Nyberg(✉), Jonas Westman, and Dilian Gurov

T. Margaria and B. Steffen (Eds.): ISoLA 2020, LNCS 12478, pp. 348–365, 2020.
https://doi.org/10.1007/978-3-030-61467-6_22

- ISoLA 2020 paper
- abstract theory based on first-order logic (FOL)
- proof system with compositionality theorem
- demonstrated on examples

Issues with Previous Work

- pen-and-paper theory and graphical proofs – trustworthy?
- theory is fully abstract, based on a set of “runs”
- previous examples are limited guide for users



Our Work: A More Trustworthy Basis

- corrected restatement of ISoLA 2020 paper theory
- machine-checked proof system soundness via HOL4 prover
- validated via John Harrison's FOL formalization
- instantiation for timed words (real-time executions)
- embedded systems decomposition example: fuel level display

Aims to pave the way for more practical applications.

Component and Specification Syntax

c:	component constant name	**S**:	specification constant name
q:	component variable	V:	specification variable

$$
\begin{aligned}
c &::= \mathbf{c} \mid c \times c \mid q \\
\mathbb{S} &::= \mathbf{S} \mid \copyright \mid T_{||} \\
S &::= \mathbb{S} \mid S \sqcap S \mid (S, S) \mid S \,||\, S \mid V
\end{aligned}
$$

- $c_1 \times c_2$ is the composition of components c_1 and c_2
- $\copyright$ specifies non-vacuousness
- $S_1 \sqcap S_2$ is the conjunction of specifications S_1 and S_2
- (A, G) is an assume-guarantee specification pair ("contract")
- $S_1 \,||\, S_2$ is parallel composition of specifications S_1 and S_2

Predicate Syntax

$$P ::= c : S \mid S \sqsubseteq S \mid \text{Assertional}(S) \mid \forall_{\mathcal{C}}\, q.\, P$$
$$\mid \forall_{\mathcal{S}}\, V.\, P \mid P \wedge P \mid \neg P \mid c =_{\mathcal{C}} c \mid S =_{\mathcal{S}} S$$

- $c : S$ means that the component c implements specification S
- $S_1 \sqsubseteq S_2$ means that specification S_1 refines specification S_2
- Assertional(S) means that S is not a hyperproperty
- $\forall_{\mathcal{C}}\, q$ quantifies over components
- $\forall_{\mathcal{S}}\, V$ quantifies over specifications
- $c_1 =_{\mathcal{C}} c_2$ asserts (behavioral) equality of components c_1, c_2
- $S_1 =_{\mathcal{S}} S_2$ asserts (behavioral) equality of specifications S_1, S_2

Semantics of Components and Specifications

We follow ISoLA 2020 paper in a theorem prover friendly way:

- Ω is an abstract set (of runs)
- components are mapped to subsets of Ω
- specifications are mapped to subsets of $\mathcal{P}(\Omega)$
- constants assigned mappings via models $\mathcal{M}$
- variables assigned mappings via substitutions σ

$$
\begin{aligned}
\llbracket \mathbf{c} \rrbracket^{\sigma}_{\mathcal{M}} &= \mathcal{M}(\mathbf{c}) \\
\llbracket c_1 \times c_2 \rrbracket^{\sigma}_{\mathcal{M}} &= \llbracket c_1 \rrbracket^{\sigma}_{\mathcal{M}} \cap \llbracket c_2 \rrbracket^{\sigma}_{\mathcal{M}} \\
\llbracket q \rrbracket^{\sigma}_{\mathcal{M}} &= \sigma(q) \\
\llbracket V \rrbracket^{\sigma}_{\mathcal{M}} &= \sigma(V)
\end{aligned}
\qquad
\begin{aligned}
\llbracket \copyright \rrbracket^{\sigma}_{\mathcal{M}} &= \{B \in \Omega \mid B \neq \emptyset\} \\
\llbracket \mathbf{S} \rrbracket^{\sigma}_{\mathcal{M}} &= \mathcal{M}(\mathbf{S}) \\
\llbracket S_1 \sqcap S_2 \rrbracket^{\sigma}_{\mathcal{M}} &= \llbracket S_1 \rrbracket^{\sigma}_{\mathcal{M}} \cap \llbracket S_2 \rrbracket^{\sigma}_{\mathcal{M}} \\
\llbracket S_1 || S_2 \rrbracket^{\sigma}_{\mathcal{M}} &= \llbracket S_1 \rrbracket^{\sigma}_{\mathcal{M}} \Cap \llbracket S_2 \rrbracket^{\sigma}_{\mathcal{M}}
\end{aligned}
$$

$$\llbracket (S_1, S_2) \rrbracket^{\sigma}_{\mathcal{M}} = \{B \mid \forall B' \in \llbracket S_1 \rrbracket^{\sigma}_{\mathcal{M}}.\ B \cap B' \in \llbracket S_2 \rrbracket^{\sigma}_{\mathcal{M}}\}$$

Semantics of Predicates (fragment)

$$
\begin{aligned}
[\![c : S]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow [\![c]\!]^{\sigma}_{\mathcal{M}} \in [\![S]\!]^{\sigma}_{\mathcal{M}} \\
[\![S_1 \sqsubseteq S_2]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow [\![S_1]\!]^{\sigma}_{\mathcal{M}} \subseteq [\![S_2]\!]^{\sigma}_{\mathcal{M}} \\
[\![\forall_{\mathcal{C}}\, q.\, P]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow \text{for all subsets } s \text{ of } \Omega,\ [\![P]\!]^{\sigma[q \mapsto s]}_{\mathcal{M}} \\
[\![\forall_{\mathcal{S}}\, V.\, P]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow \text{for all subsets } s \text{ of } \mathcal{P}(\Omega),\ [\![P]\!]^{\sigma[V \mapsto s]}_{\mathcal{M}} \\
[\![P_1 \wedge P_2]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow [\![P_1]\!]^{\sigma}_{\mathcal{M}} \text{ and } [\![P_2]\!]^{\sigma}_{\mathcal{M}} \\
[\![c_1 =_{\mathcal{C}} c_2]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow [\![c_1]\!]^{\sigma}_{\mathcal{M}} = [\![c_2]\!]^{\sigma}_{\mathcal{M}} \\
[\![S_1 =_{\mathcal{S}} S_2]\!]^{\sigma}_{\mathcal{M}} &\Leftrightarrow [\![S_1]\!]^{\sigma}_{\mathcal{M}} = [\![S_2]\!]^{\sigma}_{\mathcal{M}}
\end{aligned}
$$

Is this first-order logic (FOL)?

- it is a **two-sorted** FOL
- separate quantification for components $\mathcal{C}$ and specifications $\mathcal{S}$
- FOL domain is a disjoint union (of **sets of runs** and **sets of sets of runs**)
- translation to unsorted FOL is straightforward
- we use John Harrison's FOL formalization (FO model theory)

Example:

$$\forall_{\mathcal{C}}\, q.\, \forall_{\mathcal{S}}\, V_1.\, \forall_{\mathcal{S}}\, V_2.\, q : (V_1 \sqcap ©, V_2 \sqcap ©)$$

translates to

$$\forall\, q.\, isc(q) \rightarrow \forall\, V_1.\, isS(V_1) \rightarrow \forall\, V_2.\, isS(V_2) \rightarrow \\ impl(q, ag(conj(V_1, compat), conj(V_2, compat)))$$

FOL predicate translation soundness

Theorem

For every model $\mathcal{M}$, substitution σ, and specification language predicate P, $[\![P]\!]^{\sigma}_{\mathcal{M}}$ holds precisely when $L(\mathcal{M}), L(\sigma) \models P2f(P)$, where $\models$ is the first order satisfaction relation.

Proof system

- we extend and complete the earlier Gentzen-style proof system
- each "custom" predicate form gets intro/elim rules
- rewriting rules are necessary for doing interesting proofs
- system is a relation $\Gamma \vdash P$ with Γ set of admitted predicates

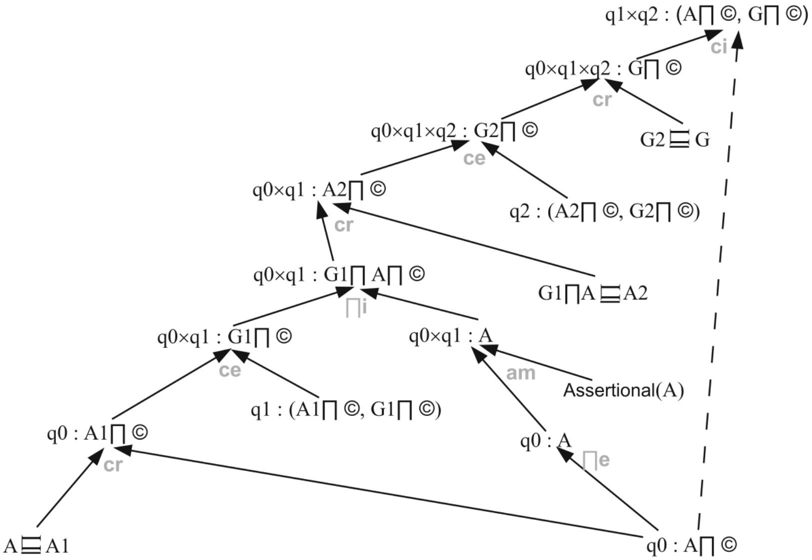
Example rules:

$$\frac{\Gamma \vdash \forall_{\mathcal{C}}\, q.\, q : S_1 \rightarrow q : S_2}{\Gamma \vdash S_1 \sqsubseteq S_2} \quad \text{REF\_IN}$$

$$\frac{\begin{array}{c}\Gamma \vdash c : S_1 \\ \Gamma \vdash S_1 \sqsubseteq S_2\end{array}}{\Gamma \vdash c : S_2} \quad \text{REF\_EL} \qquad \frac{\begin{array}{c}\Gamma \vdash c_1 : S_1 \\ \Gamma \vdash c_2 : S_2\end{array}}{\Gamma \vdash c_1 \times c_2 : S_1 || S_2} \quad \text{PAR\_IN}$$

$$\frac{\Gamma \vdash c : S_1 \sqcap S_2}{\Gamma \vdash c : S_1} \quad \text{CONJ\_EL1} \qquad \frac{\Gamma \vdash c : S_1 \sqcap S_2}{\Gamma \vdash c : S_2} \quad \text{CONJ\_EL2}$$

Graphical proof from earlier work



Proof system soundness and a corollary

Theorem

The proof system is sound with respect to the predicate semantics. That is, whenever $\Gamma \vdash P$, then for all $\mathcal{M}$ and σ, if $[\![P']\!]^{\sigma}_{\mathcal{M}}$ for all $P' \in \Gamma$, then $[\![P]\!]^{\sigma}_{\mathcal{M}}$.

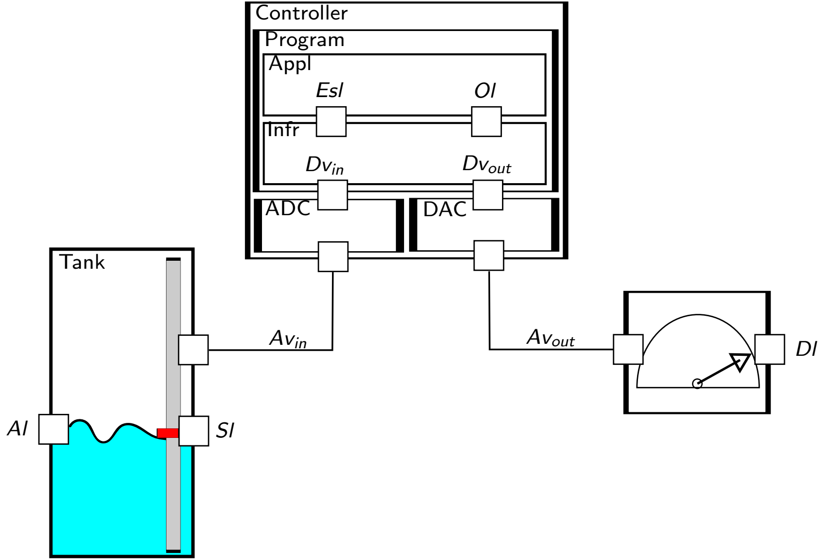
Corollary

For all $\mathcal{M}$ and σ, whenever it holds that

- $[\![S_1 || S_2 || S_3 \sqsubseteq S]\!]^{\sigma}_{\mathcal{M}}$,
- $[\![P]\!]^{\sigma}_{\mathcal{M}}$ *for every* $P \in \Gamma$,
- $\Gamma \vdash c_1 : S_1$, $\Gamma \vdash c_2 : S_2$, *and* $\Gamma \vdash c_3 : S_3$,

then $[\![c_1 \times c_2 \times c_3 : S]\!]^{\sigma}_{\mathcal{M}}$.

Application: Fuel Level Display system



Instantiation for Fuel Level Display specification

- instantiate abstract set of runs Ω to set of *timed words* Ω_{TW}
- extend specification syntax to support Metric Interval Temporal Logic (MITL) formulas
- add proof system rule for MITL
- express fuel level display system specifications using MITL

Timed words

Definition

Let $\mathcal{A}$ be a set of *system states*, and let τ be a function from natural numbers $\mathbb{N}$ to tuples $\mathcal{A} \times \mathbb{R}_{\geq 0}$. We call τ a *timed word*, and for $k \in \mathbb{N}$, we write $aval(\tau(k))$ for the first component of the tuple, the system state at k, and $tval(\tau(k))$ for the second component, representing the state's moment in time.

Extended specification syntax

$$
\begin{aligned}
I &::= [a, b] \mid (a, b] \mid [a, b) \mid [a, \infty) \mid (a, b) \mid (a, \infty) \\
\phi &::= p \mid \neg\phi \mid \phi \wedge \phi \mid \phi \,\mathsf{U}_I\, \phi' \mid \phi \,\mathsf{S}_I\, \phi' \mid \Box_I\, \phi \mid \Diamond_I\, \phi \mid \boxminus_I\, \phi \mid \diamondminus_I\, \phi \\
\mathbb{S} &::= \mathbf{S} \mid \copyright \mid T_{||} \mid \widehat{\phi}
\end{aligned}
$$

Rationale:

- define intervals using integers over non-negative reals (time)
- inject MITL formulas ϕ into specifications
- MITL formulas include formulas p on system states

MITL formula semantics (fragment)

$$t \oplus [a, b] = \{\, r \in \mathbb{R}_{\geq 0} \mid t + a \leq r \leq t + b \,\}$$
$$t \oplus (a, b] = \{\, r \in \mathbb{R}_{\geq 0} \mid t + a < r \leq t + b \,\}$$
$$t \oplus [a, b) = \{\, r \in \mathbb{R}_{\geq 0} \mid t + a \leq r < t + b \,\}$$

$$\llbracket\, p \,\rrbracket^i = \{\, \tau \in \Omega_{TW} \mid aval(\tau(i)) \models p \}$$
$$\llbracket\, \neg\phi \,\rrbracket^i = \Omega_{TW} \setminus \llbracket\phi\rrbracket^i$$
$$\llbracket\, \phi \wedge \phi' \,\rrbracket^i = \llbracket\phi\rrbracket^i \cap \llbracket\phi'\rrbracket^i$$
$$\llbracket\, \diamondminus_I\, \phi \,\rrbracket^i = \{\, \tau \in \Omega_{TW} \mid \exists j.\ tval(\tau(j)) \in tval(\tau(i)) \ominus I \wedge \tau \in \llbracket\phi\rrbracket^j \,\}$$
$$\llbracket\, \Diamond_I\, \phi \,\rrbracket^i = \{\, \tau \in \Omega_{TW} \mid \exists j.\ tval(\tau(j)) \in tval(\tau(i)) \oplus I \wedge \tau \in \llbracket\phi\rrbracket^j \,\}$$

$$\llbracket\widehat{\phi}\rrbracket^{\sigma}_{\mathcal{M}} = \mathcal{P}(\llbracket\phi\rrbracket^0)$$

Fuel Level Display Specification in MITL

MITL formula ϕ_{FLD}:

$$\Box_{[0,\infty)} (Dl = r \to \diamondminus_{[0,t]} (Al \approx_m r)).$$

- Dl is **displayed** level of fuel
- Al is **actual** level of fuel
- $x \approx_m v$ is a shorthand for the absolute difference of x and v being less or equal to a predetermined margin of error m in some unit of time

Fuel Level Display System Decomposition

$$c_{FLD} = c_{meter} \times c_{ctrl} \times c_{tank}.$$

$$\begin{aligned}
\phi_{meter} &: \Box_{[0,\infty)} \left(Dl = f(v) \rightarrow \Diamond_{[0,t_1]} \left(A_{v_{in}} \approx_m v \right) \right) \\
\phi_{ctrl} &: \Box_{[0,\infty)} \left(A_{v_{in}} \approx_m v \rightarrow \Diamond_{[0,t_2]} \left(A_{v_{out}} \approx_m v \right) \right) \\
\phi_{tank} &: \Box_{[0,\infty)} \left(A_{v_{out}} \approx_m v \rightarrow \Diamond_{[0,t_3]} \left(Al \approx_m f(v) \right) \right)
\end{aligned}$$

Via corollary, decomposition requires:

- $S_{meter} || S_{ctrl} || S_{tank} \sqsubseteq S_{FLD}$, where $S_{FLD} = \widehat{\phi}_{FLD}$
- $c_{meter} : S_{meter}$ where $S_{meter} = \widehat{\phi}_{meter}$, use proof system.
- $c_{ctrl} : S_{ctrl}$ where $S_{ctrl} = \widehat{\phi}_{ctrl}$, use proof system.
- $c_{tank} : S_{tank}$ where $S_{tank} = \widehat{\phi}_{tank}$, use proof system.

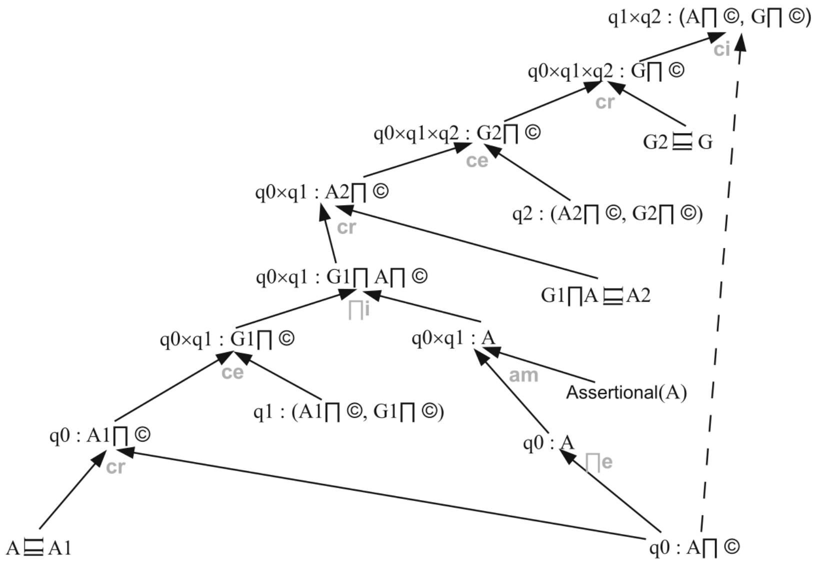
HOL4 formalization approach

1. encode abstract syntax and proof system in Ott metalanguage
2. annotate Ott code for translation to HOL4
3. export Ott code to HOL4 code
4. formulate metatheory (semantics, theorems) inside HOL4
5. prove theorems (type `thm`) using proof tactics

```
c :: c_ ::=
 {{ com component term }}
| cn :: :: const
  {{ com constant }}
| c * c' :: :: comp
  {{ com composition }}
| q :: :: var
  {{ com variable }}
```

```
val _ = Hol_datatype `
c =
(* component term *)
   c_const of cn
 | c_comp of c => c
 | c_var of q
`;
```

Graphical proof from earlier work again



Is the graphical proof correct?

Strictly speaking, no.

- the proof is modulo associativity/commutativity of operators
- information is missing on how rules should be applied
- formal proof in HOL4 uses rewriting *inside* system

HOL4 source fragment

```
Theorem example_refine_spec_holds:
 !A A1 A2 G G1 G2.                     (* metavariables *)
 spec_holds                            (* binary relation *)
  (example_Ps A A1 A2 G G1 G2)         (* premises *)
  (example_goal A A1 A2 G G1 G2)       (* goal *)
Proof
 rw [example_goal] >>

 MATCH_MP_TAC all_in_c >>
 Q.EXISTS_TAC `"q1"` >>
 (* .... *)
QED
```

Conclusions

- theory of specifications validated and checked in HOL4
  - connection to FOL
  - proof system soundness
  - practical proofs fully checked inside system
  - around 6000 lines of open source code
- instantiation for timed words and fuel level display decomposition
- corrected many minor(?) issues, in particular sortedness

`https://github.com/rse-verification/contract-compositionality`

Ongoing and Future Work

- Coq/Rocq version finished (exported from Ott, no proofs)
- ongoing adaptation in Coq/Rocq for probabilistic domain
  - pen-and-paper version published
  - master's project formalizes in Coq/Rocq
- textual proof format and practical checker (via CakeML?)