Formal Analysis for Communicating Medical Devices

Mike Whalen
Program Director, UMSEC
University of Minnesota

Research Topics

M. Whalen, A. Gacek, and D. Cofer. **Hierararchical Circular Compositional Reasoning.** UMSEC Tech Report 2012-1

- Multi-Domain Analysis of System Architecture Models
  - Compositional Assume-Guarantee Reasoning
  - Next: Incorporating different notions of time

D. Cofer, A. Gacek, S. Miller, M. Whalen, and B. LaValley. **Compositional Verification of Architectural Models.** ***NFM 2012***

W. Visser, M. Dwyer, and M. Whalen, **The Hidden Models of Model Checking.** ***Journal of Software and Systems Modeling.*** *[Submitted – Under Review]*

- Analysis of Component-Level MBD Models
  - Simulink/Stateflow
  - Rhapsody

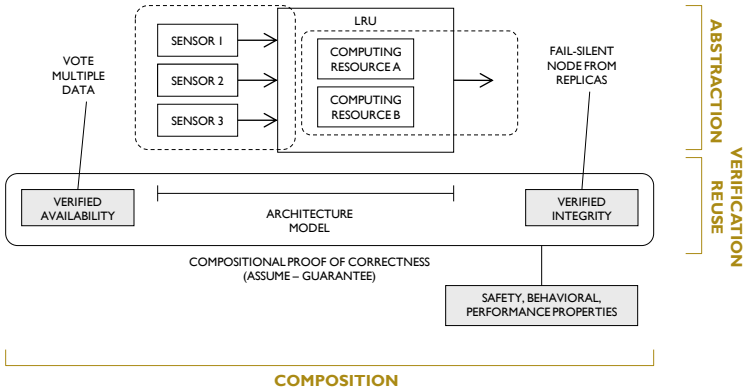
T. Kahsai, P.L. Garoche, C. Tinelli, and M. Whalen. **Incremental Verification with Mode Machine Invariants in State Machines.** ***NFM 2012***

- Automated Analysis of Datatype-Manipulating Programs
  - Automated proofs of (arbitrarily large) data structures.
  - Based on extension of Kuncak & Suter POPL11 algorithm

D. Hardin, K. Slind, M. Whalen, and T.H. Pham. **The Guardol Language and Verification System,** ***TACAS 2012***

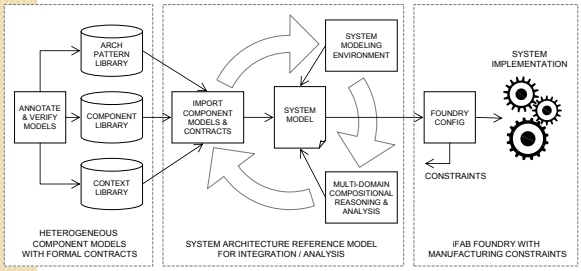
Vision

System design & verification through pattern application and compositional reasoning



Multi-domain modeling & analysis

- Architecture model captures properties, relationships, contracts between and within domains
- Supports system-level analysis



Behavioral
(Subset of) PSL properties

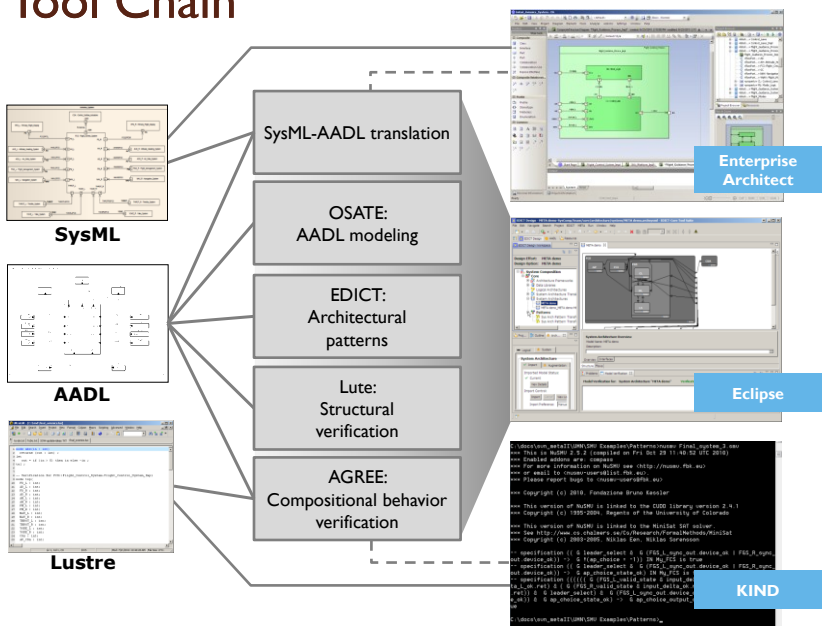
Structural/Static
Model constraints

Probabilistic
Fault modeling

Resource Allocation
Schedule, memory, bandwidth

Manufacturability
Resources, processes

Tool Chain



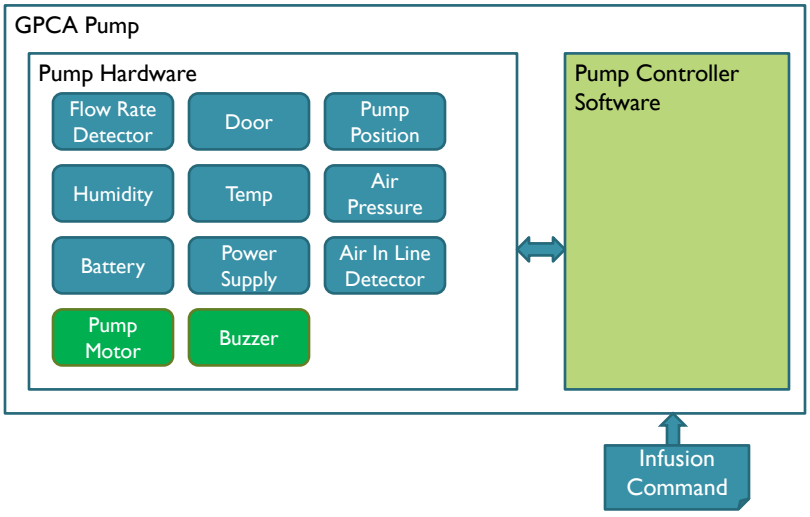
5

GPCA Pump Example

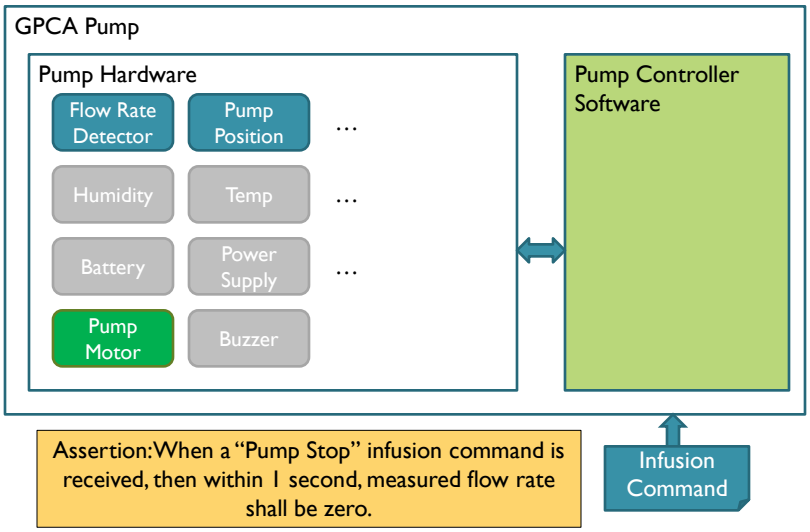
- Property of Interest:
  - If a "Pump Stop" command is received, then within 1 second, measured flow rate shall be zero.
- We will prove this property compositionally based on the architecture of the Pump subsystem.

With exciting verification demo!

Architecture of GPCA Pump

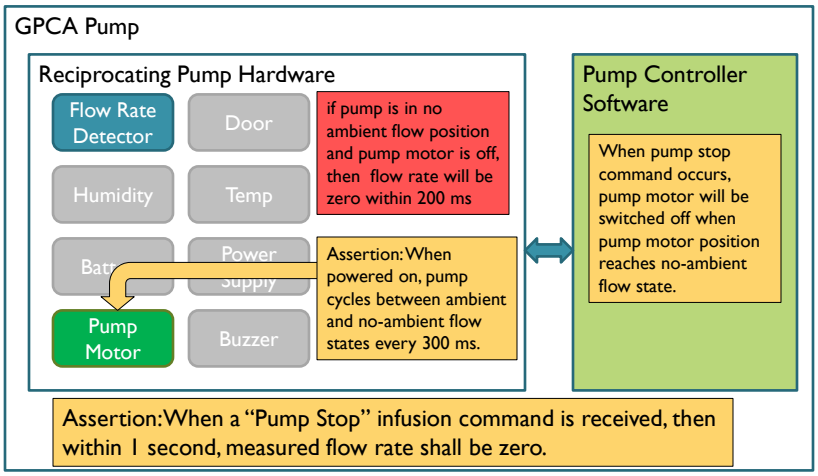


Proof of GPCA Pump

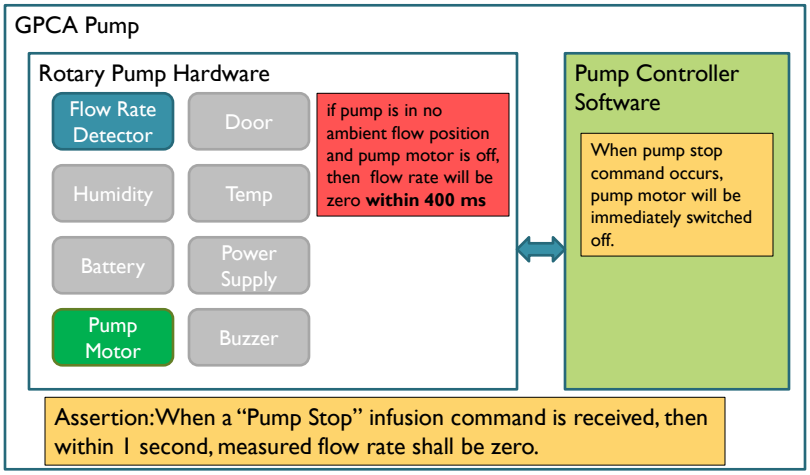


Assertion: When a "Pump Stop" infusion command is received, then within 1 second, measured flow rate shall be zero.

Proof of Reciprocating Pump



Rotary Pump



DEMO

Underlying Formalism: Circular Compositional Reasoning

- Suppose we have
  - *Sets of formulas* Γ *and* Q
  - *A well-founded order* $\prec$ *on* Q
  - *Sets* $\Theta_q \subseteq \Delta_q \subseteq Q$, *such that* $r \in \Theta_q$ *implies* $r \prec q$
- Then if for all q ∈ Q
  - $\Gamma \Rightarrow G((Z(H(\Theta_q)) \wedge \Delta_q) \Rightarrow q)$
- Then:
  G(q) for all q ∈ Q
- [Adapted from McMillan]

Formulation applied to Hierarchical Reasoning

- So given component contracts:
  $\Gamma = \{ G(H(A_c) \Rightarrow P_c) \mid c \in C \}$
- …we add a set of obligations that tie the system assumption to the component assumptions

$$Q = \blacksquare \cup \{H(A_s) \implies A_c \mid c \in C\}$$

- We can prove G(q) for all elements of Q
- …which means we prove our system property

Problem: Liveness

- Obligations are of the form:
  $\blacksquare \Rightarrow G((Z(H(\Theta_q)) \wedge \Delta_q) \Rightarrow q)$
  where $\Gamma = \{ G(H(A_c) \Rightarrow P_c) \mid c \in C \}$
- Unfortunately, having G operator on the left-hand of an implication means that this is a liveness formula.
  - We want to use provers that only support safety
- We want to reflect the component guarantees directly into the G operator on the right.

Safety Formulation

- We define *c*^ as (A_c ^ P_c) for component *c*.
  - …and define C^ as {*c*^ | c ∈ C }
  - …and define Δ_c and Θ_c to match Δ_c and Θ_q.
- Then we can define obligations that involve only past-time temporal operators:

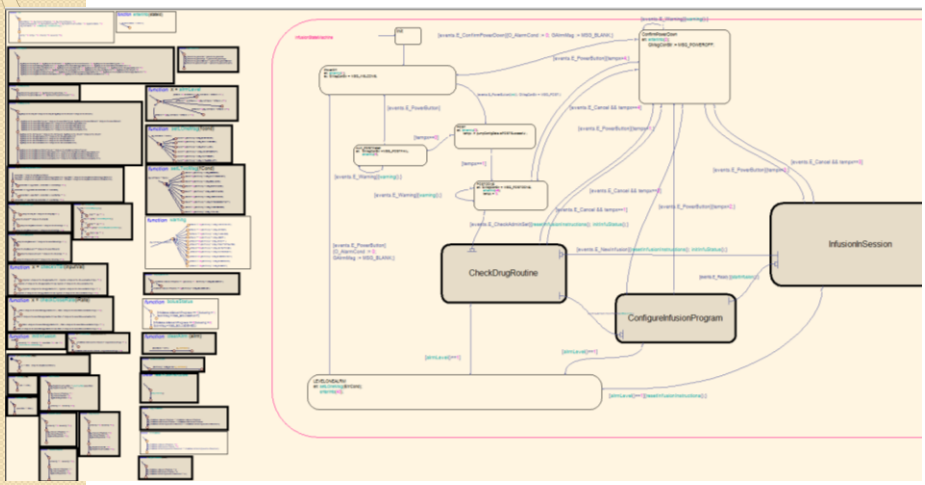
$$((\forall c \in C : \ (\sigma \vdash G((H(A_s) \wedge Z(H(\Delta_c)) \wedge \Theta_c) \implies A_c))) \wedge (\sigma \vdash (G((H(A_s) \wedge H(C^\wedge)) \implies P_s))))$$

- …to establish ($G(H(A_s)) \Rightarrow P_s$)

D. Cofer, A. Gacek, S. Miller, M. Whalen, and B. LaValley. **Compositional Verification of Architectural Models.** ***NFM 2012***

M. Whalen, A. Gacek, and D. Cofer. **Hierararchical Circular Compositional Reasoning.** UMSEC Tech Report 2012-1

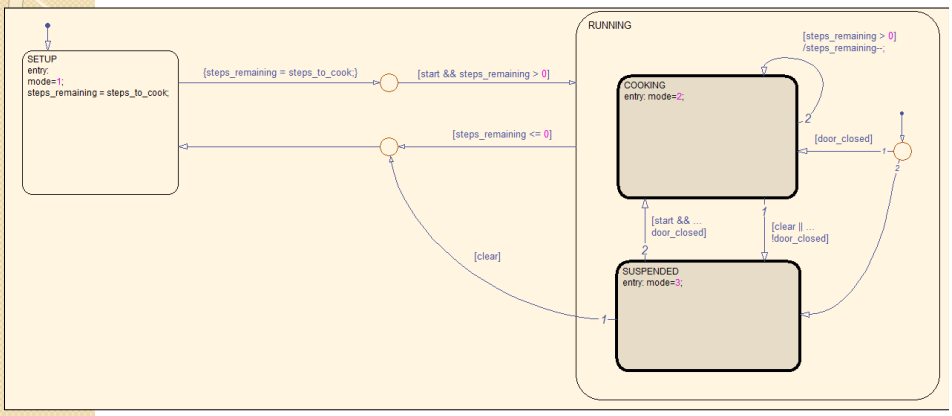
Component-Level Analysis



Making k-induction model checkers mode aware

- K-induction is a model checking technique that can be used with SMT solvers
  - **Very scalable** if properties can be **inductively proven**
  - Unfortunately, Inductive proofs **often fail** because properties are **too weak**
  - Lots of work on lemma/invariant discovery to **strengthen** properties
    - Bjesse and Claessen: *SAT-based verification without State Space Traversal*
    - Bradley: *SAT-based Model Checking without Unrolling*
    - Tinelli: *Instantiation-Based Invariant Discovery*
  - However, these techniques do not work for state machines / modes
- Created new lemma discovery technique for modes and implemented it in Kind model checker
  - Discover cliques of integer or enumerated model variables
    - Use abstract interpretation to discover small subrange integer modes
  - Posit relationships between mode variables and inductively verify.
  - Initial results are very positive [NFM 2012]

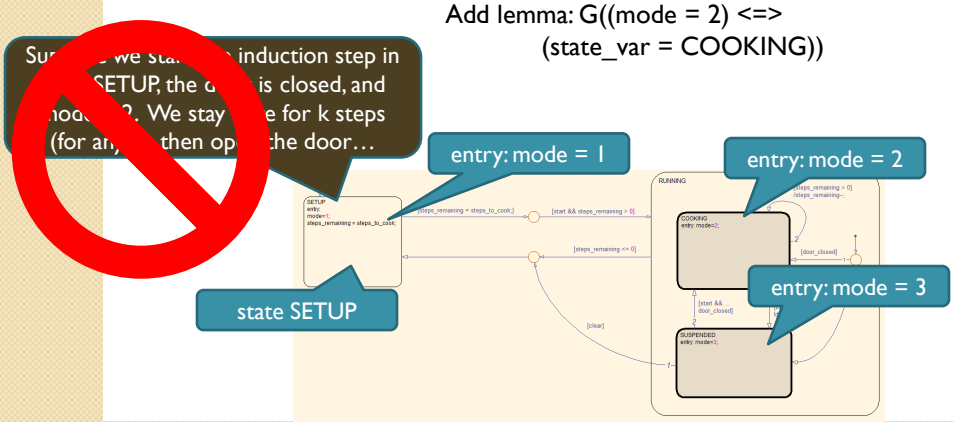
Simple Example of Induction Failure: Microwave Mode Logic



- Want to prove if we're cooking then the door is closed
  G((mode = 2) => door_closed);

Making it inductive

- Induction step: Start from **arbitrary state** in which property holds: G((mode = 2) => door_closed);
- Note that the mode variable does not directly affect the value of the state machine.
- Need to relate entry concrete value to state machine value.
  Add lemma: G((mode = 2) <=>
  (state_var = COOKING))



GPCA Pump Simulink/Stateflow Model

- Simulink/Stateflow GPCA pump controller
  - Generic Patient-Controlled Analgesia
    - Infusion pump with input from the patient
  - Reference model for model-based development for medical devices
- Analysis through test-case generation (Reactis)
- Analysis through model checking
  - Kind and SAL using RCI/UMN Gryphon tool set

Conclusion

- Mature Simulink/Stateflow analysis capability
  - *Gryphon* tool suite and Kind model checker
  - Ongoing high-visibility projects at Rockwell Collins using model checking (CAS: Crew Alerting System)
  - Recent capabilities in Kind make it significantly stronger tool
  - Using this to analyze large GPCA models
- New AGREE system architecture analysis capability
  - Support models in AADL and SysML
  - Tools built in Eclipse – Freely available
  - Translates to Kind and will eventually target more: (NuSMV, PVS)
- Combining results from several funding streams
  - Kind invariant work co-sponsored by AFOSR
  - AGREE work co-sponsored by DARPA (META-II program)
- Creating substantial reasoning capabilities for tools that engineers *use!*