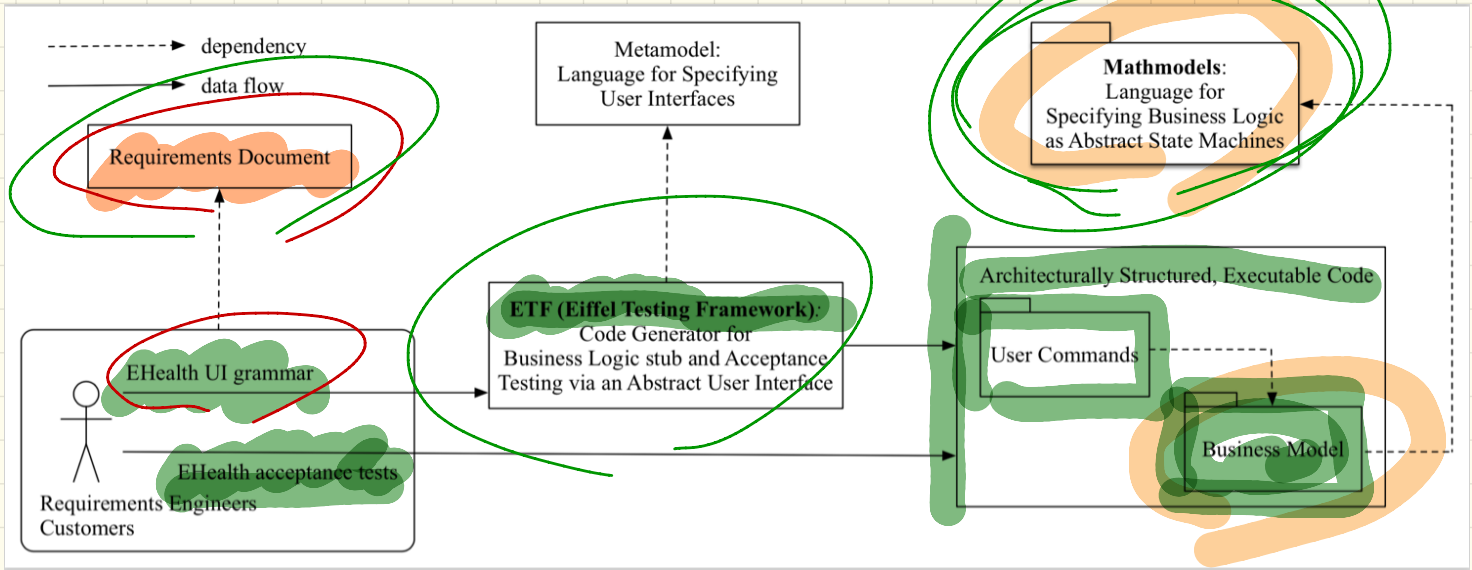
MoDRE'18

Modelling and Testing Requirements via Executable Abstract State Machines

Research Contributions



Abstract User Interface

```
system ehealth
  -- semantics types
  type MEDICATION = STRING
  type PATIENT = STRING
  -- events
  add_patient          (p: PATIENT)
  add_medication       (m: MEDIACTION)
  add_interaction      (m1: MEDIACTION; m2: MEDIACTION)
  add_prescription     (p: PATIENT; m: MEDIACTION)
  remove_interaction   (m1: MEDIACTION; m2: MEDIACTION)
  remove_prescription  (p: PATIENT; m:MEDIACTION))
```

Abstract State Variables

$$
\begin{aligned}
&\textit{patients} && \in && \mathbb{P}\,\textit{PATIENT} \\
&\textit{medications} && \in && \mathbb{P}\,\textit{MEDICATION} \\
&\textit{interactions} && \in && \textit{MEDICATION} \leftrightarrow \textit{MEDICATION} \\
&\textit{prescriptions} && \in && \textit{PATIENT} \leftrightarrow \textit{MEDICATION}
\end{aligned}
$$

Acceptance Test

```
...
 state 16
 patients:      {p1,p2,p3}
 medications:   {m1,m2,m3,m4}
 interactions:  {m1->m2,m2->m1,m2->m4,m4->m2}
 prescriptions: {p1->m1,m3; p3->m2}
->add_prescription("p3","m4")
 state 17 Error e4: this prescription dangerous
->remove_interaction("m2","m4")
 state 18
 patients:      {p1,p2,p3}
 medications:   {m1,m2,m3,m4}
 interactions:  {m1->m2,m2->m1}
 prescriptions: {p1->m1,m3; p3->m2}
->add_prescription("p3","m4")
 state 19
 patients:      {p1,p2,p3}
 medications:   {m1,m2,m3,m4}
 interactions:  {m1->m2,m2->m1}
 prescriptions: {p1->m1,m3; p3->m2,m4}
```

From Requirements to Safety Invariants

ENV3	If one *medication* interacts with another, then the reverse also applies (Symmetry).
ENV4	A medication does not interact with itself (Irreflexivity).

REQ6	The system maintains records of *patient prescriptions*. No prescription may have a *dangerous interaction*.

```
class
  HEALTH_SYSTEM
feature -- abstract state
  patients: SET [PATIENT]
  medications: SET [MEDICATION]
  prescriptions: REL [PATIENT, MEDICATION]
  interactions: SET [INTERACTION]
invariant
  symmetry_ENV3:
    across medications as m1 all
    across medications as m2 all
      interactions.has ( [m1.item, m2.item] ) = interactions.has ( [m2.item, m1.item] )
    end end
  irreflexivity_ENV4:
    across medications as m1 all not interactions.has ( [m1.item, m1.item] ) end
  no_dangerous_interactions_REQ6:
    across prescriptions.domain as p all
    across prescriptions [p.item] as m1 all
    across prescriptions [p.item] as m2 all
      interactions.has ( [m1.item, m2.item] )
        implies not( prescriptions.has( [p.item,m1.item] ) and prescriptions.has( [p.item,m2.item] ) )
    end end end
  consistent_domain:
    prescriptions.domain ⊆ patients
end
```

From Requirement to Command

REQ7	Physicians shall be allowed to add a medication to a patient's prescription, provided it does not result in a dangerous interaction.

```
class
  ADD_PRESCRIPTION
inherit
  HEALTH_SYSTEM -- inherits all system invariants
feature -- commands
  add_prescription (p: PATIENT; m: MEDICATION)
        -- Add a prescription of 'm1' to 'p1'.
    require
      -- p ∈ patients
      patients.has (p)
      -- m ∉ prescriptions[p]
      not prescriptions[p].has (m)
      -- cannot cause a dangerous interaction
      -- ∀ med ∈ prescriptions[p] : (med, m) ∉ interaction
      across prescriptions[p] as med all not interactions.has( [med.item, m] ) end
    do
      prescriptions[p].extend ([p, m])
    ensure
      prescriptions ~ old prescriptions + [p, m]
      -- UNCHANGED (patients, medications, interactions)
    end
end
```