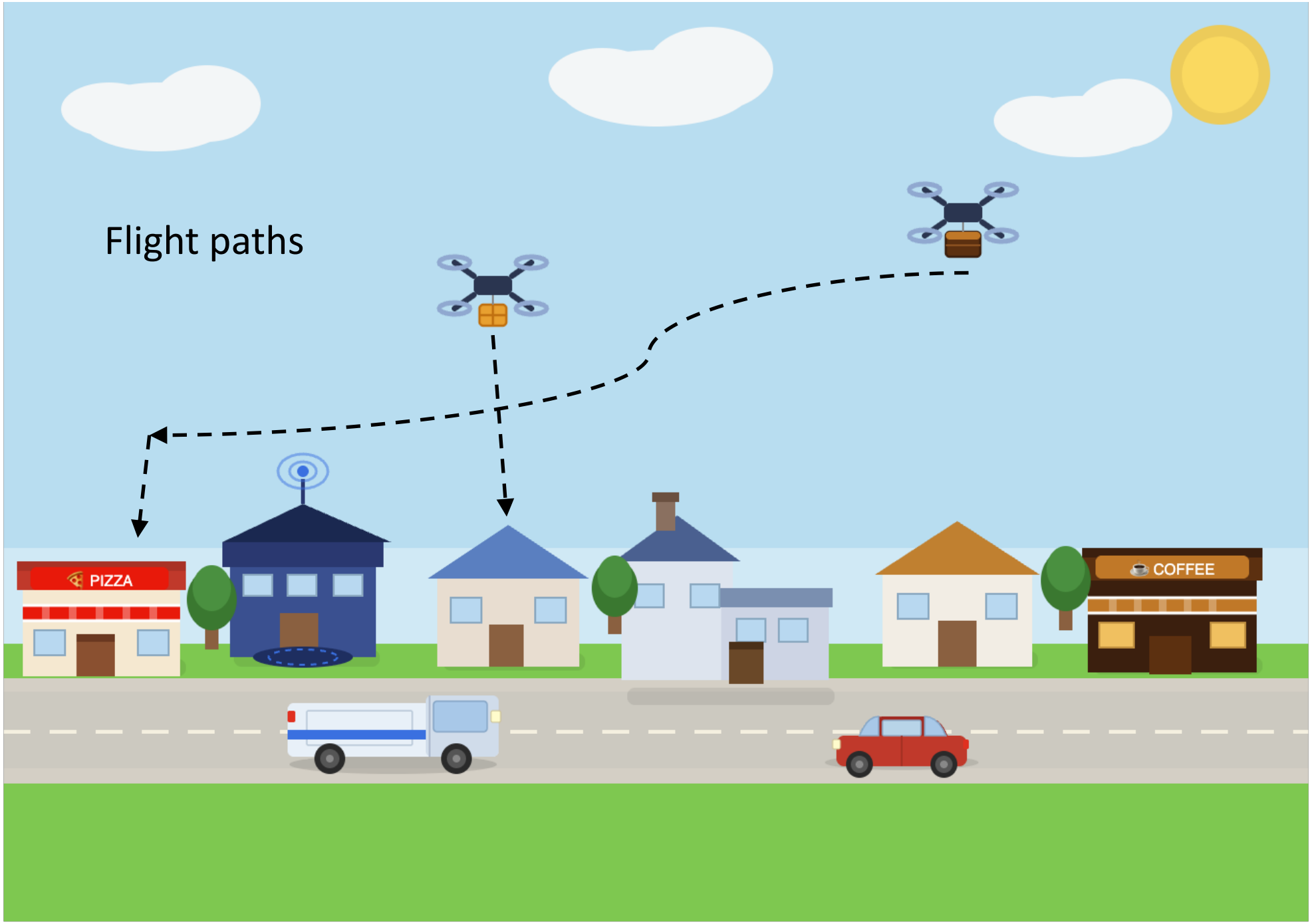
University of Southampton

Abstraction and refinement of continuous behaviour in cyber physical system

Michael Butler

WG 2.3 Meeting, Chilworth, September 2026

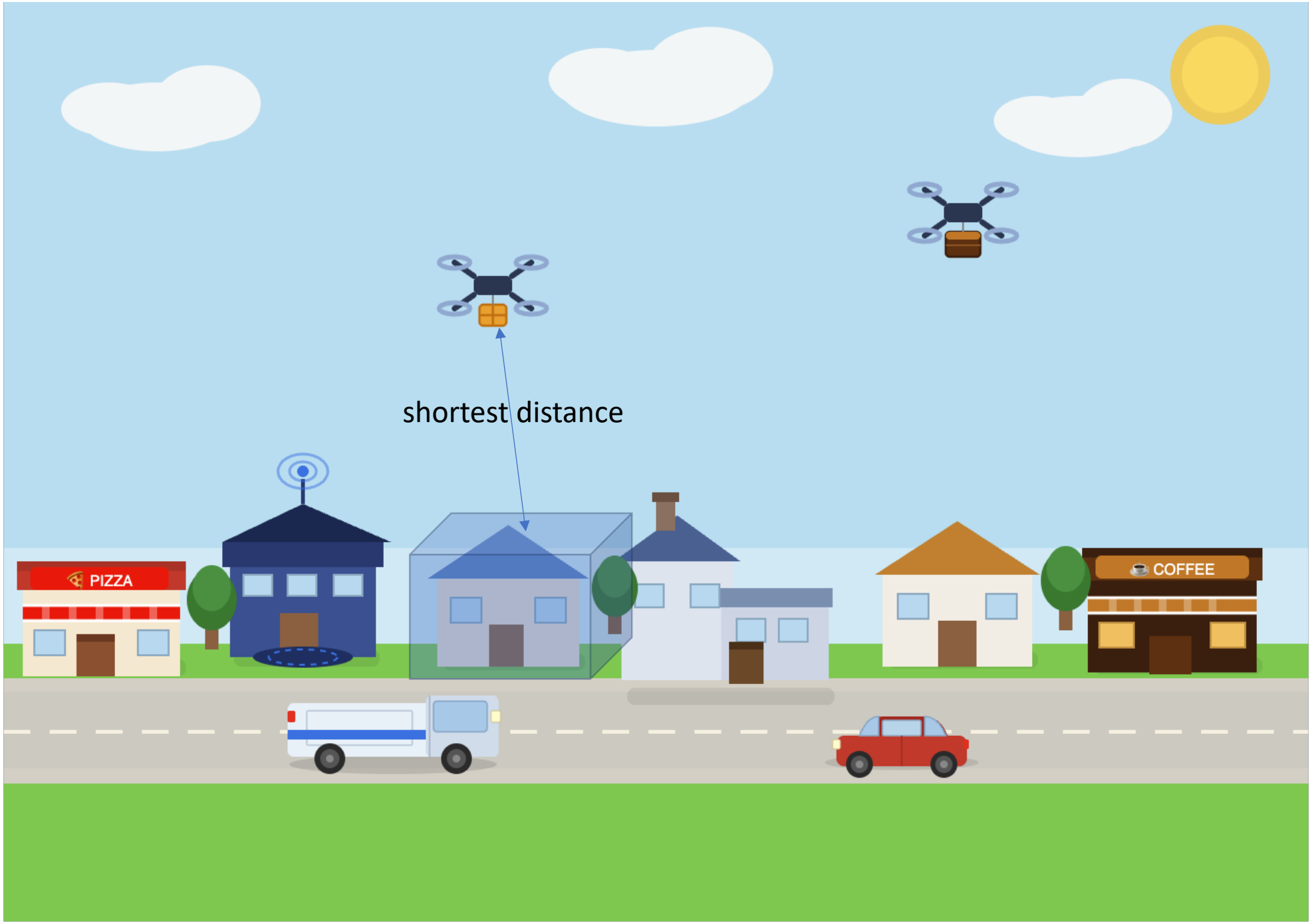

Flight paths
PIZZA
COFFEE

Conflict free paths for drones

- Path: continuous function of time to 3-D space

- Two paths are conflict-free when at all times, the distance between them is at least the minimum separation (ms)

$$\text{conflict\_free}(\, p1, p2, ms \,) \triangleq$$
$$\forall\, t \cdot t \in \text{TIME} \Rightarrow ms \leq \text{dist}(\, p1(t), p2(t) \,)$$


shortest distance
PIZZA
COFFEE

Conflict free path and object

Given

Path p

3D object o

ms minimum separation

conflict_free(p, o, ms) ≜

∀ t· t ∈ TIME ⇒ ms ≤ shortest_distance(p(t), o)

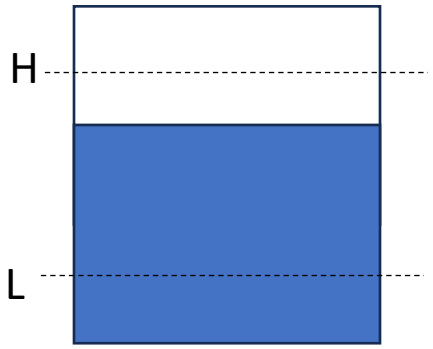
University of Southampton

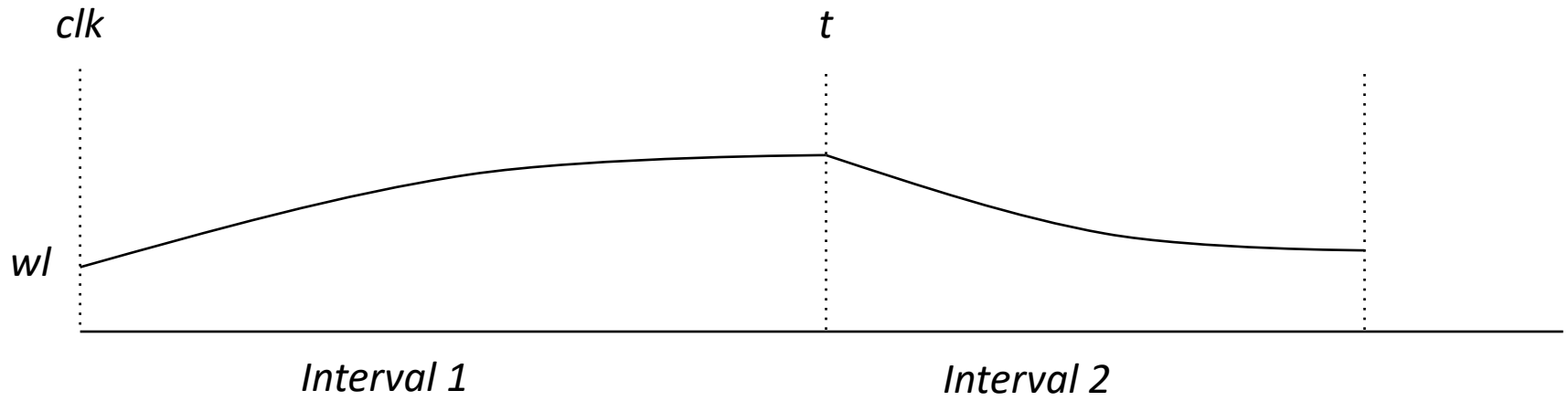
Event-B: Machines and Refinement

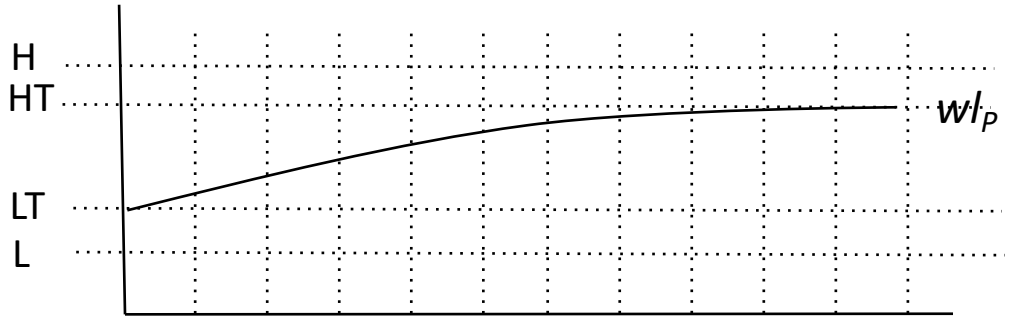
- Machine M1
  - Variables
  - Invariants
  - Collection of guarded atomic nondeterministic events

- Refined Machine M2 refines M1
  - Variable that may data-refine abstract variables
  - (Gluing) Invariants
  - Each event either
    - Refines an event of M1, or
    - Refines skip (new event)

In a nutshell: piecewise refinement of abstract continuous water level in water tank







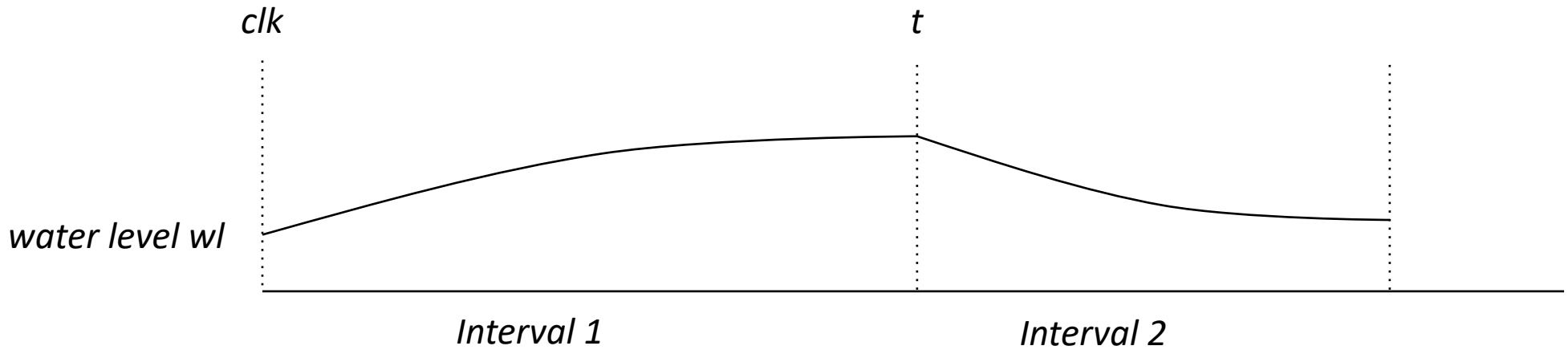
Periodic intervals duration P

Modelling Continuous Behaviour in Event-B

- MJ Butler, JR Abrial, R Banach, *Modelling and Refining Hybrid Systems in Event-B and Rodin*, From Action Systems to Distributed Systems, 2016

- variable clk ∈ $\mathbb{R}^+$
- variable wl ∈ $\mathbb{R}^+ \pfun \mathbb{R}$ // partial function
- invariant wl ∈ cts(0,clk) // wl is a continuous function
- invariant ran(wl) ⊆ L..H // wl is bounded

- Invariants are used to specify a property satisfied by the entire evolution of continuous variables.
- Interval events are used to specify a property on continuous evolution within an interval of nondeterministic duration.



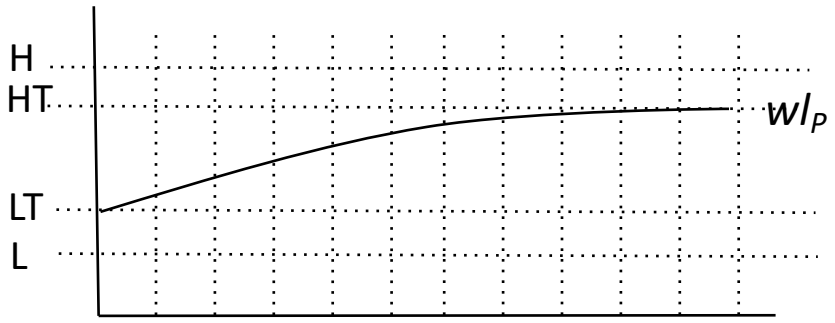
invariant ran(wl) ⊆ L..H

axiom

∀f • f ∈ cts(i,j)
f(i), f(j) ∈ L..H
f ∈ mono_inc
⇒
ran(f) ⊆ L..H

event IncreaseWaterLevelInterval
any t,f where
t ≥ clk+ε
f ∈ cts(clk,t)
f(clk) = wl(clk)
f ∈ mono_inc
f(t) ∈ L..H
then
wl := wl ∪ f
clk := t
end

Refine by periodic control of pump



Periodic intervals duration P

invariants

pump =on ⇒ $wl_P \in$ mono_inc

pump =off ⇒ $wl_P \in$ mono_dec

axioms

$L < LT < HT < H$

$L \leq LT - (RD \times T)$

$HT + (RI \times P) \leq H$

RI: rate of increase when pump on

new event PeriodicIntervalIncrease

any f_P where

pump = on

$wl_P(clk_P) \leq HT$

$f_P \in cts(clk_P , clk_P + P)$

$f_P(clk_P) = wl_P(f_P)$

$f_P(clk_P + P) \leq wl_P(clk_P) + (RI \times P)$

$f_P \in$ mono inc

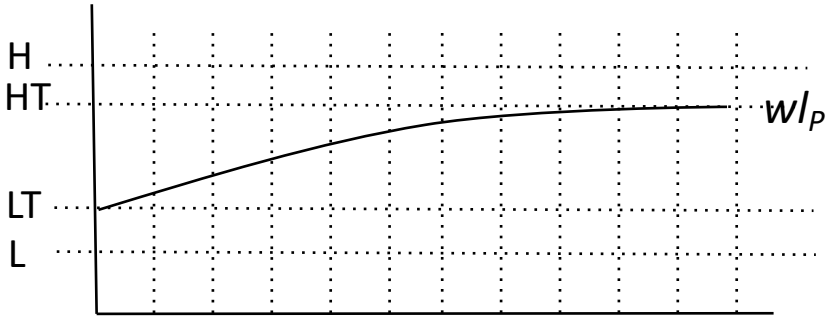
then

$wl_P := wl_P \cup f_P$

$clk_P := clk_P + P$

end

Refining the abstract interval event



Periodic intervals duration P

refined event IncreaseWaterLevelInterval

any f_P where

pump = on

$wl_P(clk_P) > HT$

then

$wl := wl \cup wl_P$

$wl_P := \{clk_P\} \triangleright wl_P$

$clk := clk_P$

pump := off

end

Refine: merge big and small step variables

invariant $wl_M = wl \cup wl_P$

invariant $clk_M = clk_P$

- Big step interval simplifies to:

refined event IncreaseWaterLevelInterval
pump = on
$wl_M(clk_M) > HT$
then
pump := off
end

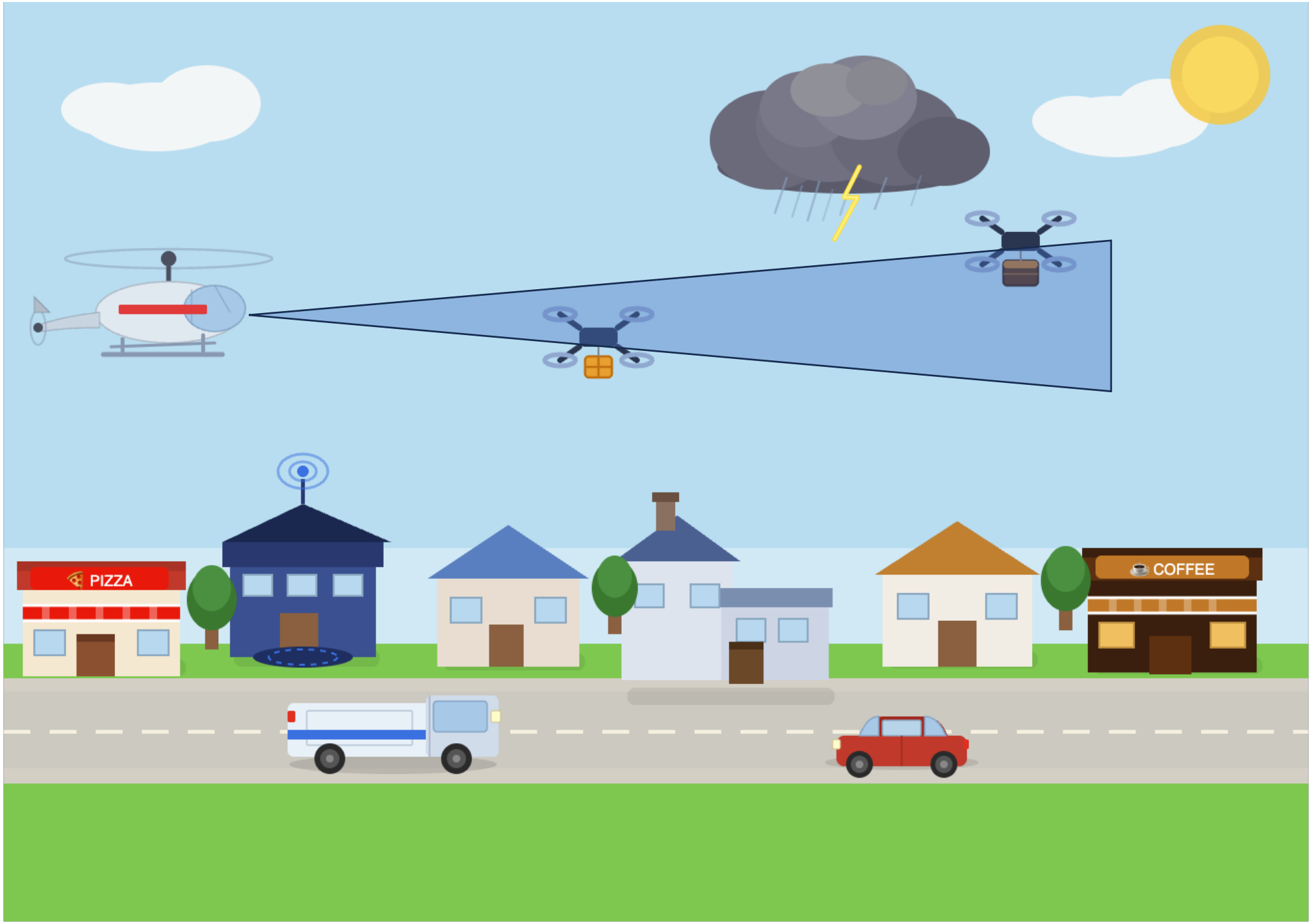
University of Southampton

Refining segment interval events

- Periodic control of pump: introduce new events to break large nondeterministic continuous intervals to smaller period intervals
- Effect of this refinement is to resolve abstract nondeterminism in earlier stages
- This allows to model actions that prevent violation of constraints on abstract continuous intervals, e.g.,
  - Choose path that avoids known static obstacles
  - Change path if dynamic object detected and predicted to lead to conflict

Safe flight path with (nearby) static objects

Event FlySegment
any t,s where
t ≥ clk+ε
s ∈ cts^3(clk,t) \\ cts in 3 dimensions
s(clk) = path(clk)
∀o • o ∈ near_obs ⇒ conflict_free(s,o)
then
path := path ∪ s
clk := t
end


PIZZA
COFFEE

Adding dynamic (nearby) objects

Event FlySegment
any t, s, vs, near_vehs **where**
t ≥ clk+ε
s ∈ cts^3(clk,t)
s(clk) = path(clk)
near_vehs ⊆ vehicles
vs ∈ near_vehs → cts^3(clk,t)
∀o • o ∈ near_objs ⇒ conflict_free(s,o)
∀v • v ∈ near_vehs ⇒ conflict_free(s,vs(v))
then
path := path ∪ s
vpaths(v) := vpaths(v) ∪ vs(v), **all** v ∈ near_vehs
clk := t
end

University of Southampton

Refining segment interval events

- Periodic control of pump: introduce new events to break large nondeterministic continuous intervals to smaller period intervals
- Effect of this refinement is to resolve abstract nondeterminism in earlier stages
- This allows to model actions that prevent violation of constraints on abstract continuous intervals, e.g.,
  - Choose path that avoids known static obstacles
  - Change path if dynamic object detected and predicted to lead to conflict

Refinement of paths: Waypoints, routes, segments

- Waypoint: time and 3-D position
- Route: sequence of waypoints increasing in time
- Segment: straight line path between successive waypoints on a route
- C Bogdiukiewicz, M Butler, TS Hoang, M Paxton, J Snook, X Waldron, T Wilkinson, *Formal development of policing functions for intelligent systems,* International Symposium on Software Reliability Engineering (ISSRE) 2017

Refinement: new periodic event

new event PeriodicSafeContinue
any s_p where
s_p is a collision free extension of path with duration P
then
$path_p := path_p \cup s_p$
$clk_p := clk_p + P$
end

Different Refinements of FlySegment

Event NextWaypoint
Refines FlySegment
when
next waypoint reached
then
target := next waypoint
end

Event ChangeDirection
Refines FlySegment
when
potential collision detected
then
target := new safe waypoint
end

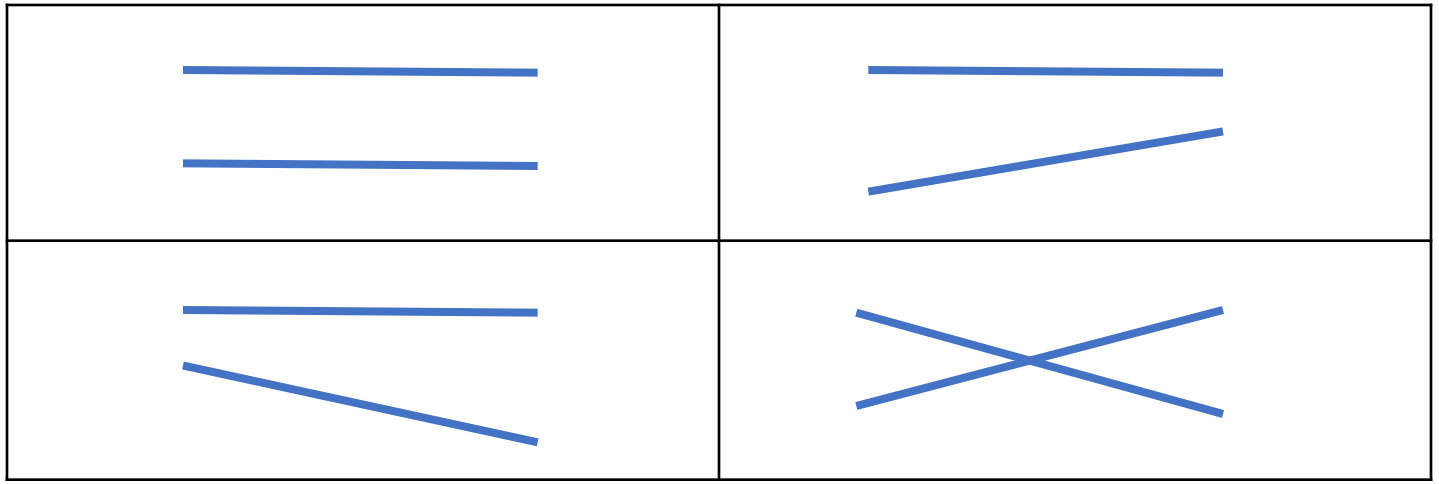
Event Hover
Refines FlySegment
when
potential collision detected
safe to hover
then
activate hover
end

Checking linear segments

- Abstract spec involves universal quantification over real interval:

$$\forall\, t \cdot\ t \in \mathrm{TIME} \ \Rightarrow\ ms \leq dist(\ p1(t),\ p2(t)\ )$$

- Refined to geometric argument on linear segments



Guaranteed Safe AI system

1. The ***probabilistic safety specification*** encodes societal risk criteria, which should ideally be determined by collective deliberation.
2. The ***verifier*** provides a quantitative guarantee … that the AI system satisfies the specification with respect to the ***world model***.
3. All potential future effects of the AI system and its environment **relevant to the safety specification** should be modelled (and – if feasible – conservatively **overapproximated**) by the world model.

- *Towards Guaranteed Safe AI: A Framework for Ensuring Robust and Reliable AI Systems*
  Dalrymple, Skalse, Bengio, Russell, Tegmark, Seshia, Omohundro, Szegedy, Goldhaber, Ammann, Abate, Halpern, Barrett, Zhao, Tan, Wing, Tenenbaum, 2024

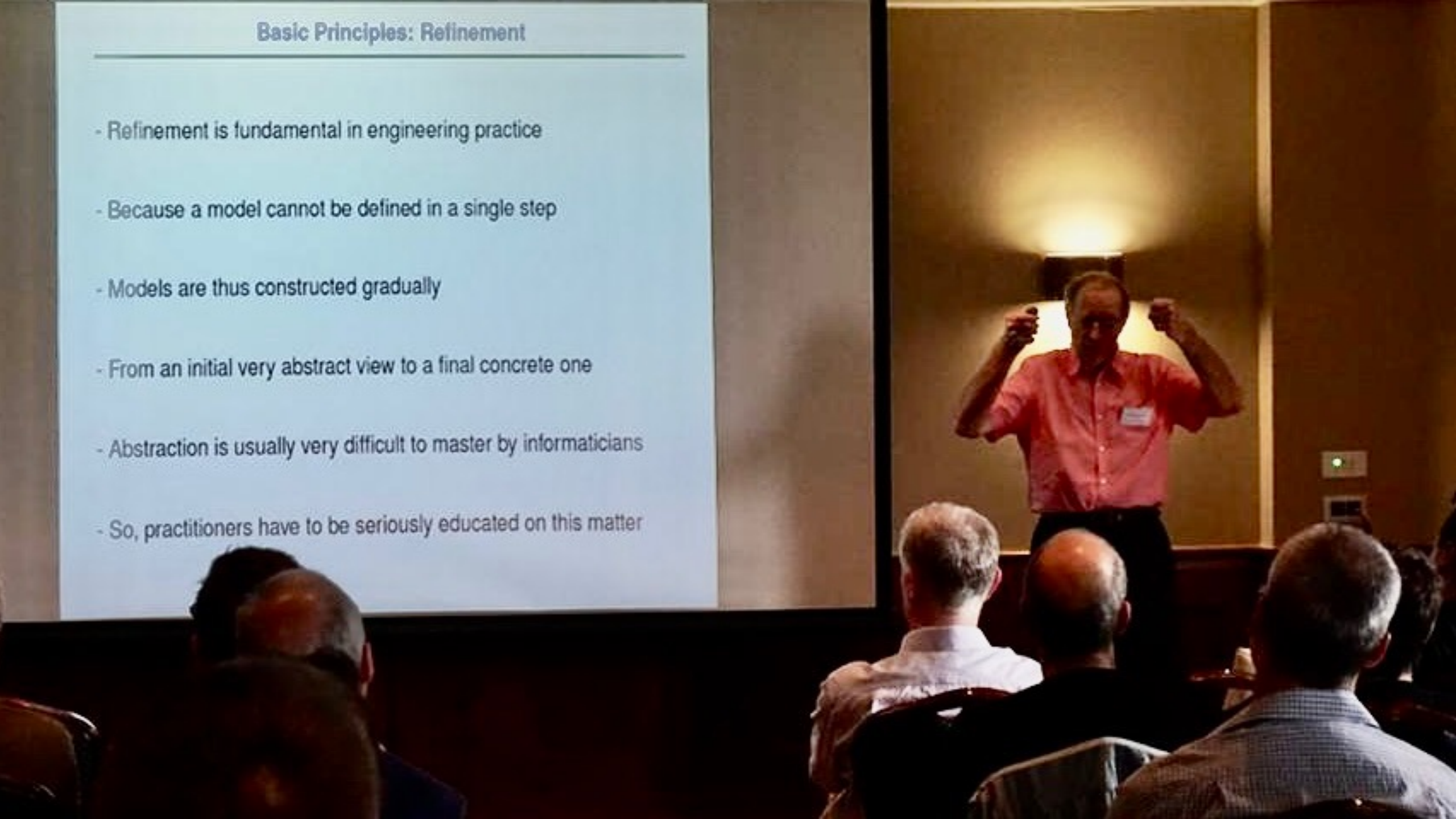
Modeling vs. programming

Programming is the activity of constructing a piece of formal text that is supposed to instruct a computer how to fulfil certain tasks. *Our intention is not to do that.* What we intend to build is a system within which there is a certain piece of software (the one we shall construct), which is a component among many others. This is the reason why our task is not limited to the software part only.

In doing this as engineers, we are not supposed to instruct a computer; rather, we are supposed to instruct ourselves. To do this in a rigorous way, we have no choice but to build a complete *model* of our future system, including the software that will eventually be constructed, as well as its environment, which, again, is made of equipment, varying physical phenomena, other software, and even users. Programming languages are of no help in doing this. All this has to be carefully modeled so that the exact assumptions within which our software is going to behave are known.

Modeling is the main task of system engineers. Programming then becomes a subtask which might very well be performed automatically.

Consequently, what has been done in the past (and still is) with the help


Basic Principles: Refinement
- Refinement is fundamental in engineering practice
- Because a model cannot be defined in a single step
- Models are thus constructed gradually
- From an initial very abstract view to a final concrete one
- Abstraction is usually very difficult to master by informaticians
- So, practitioners have to be seriously educated on this matter

Guaranteed Safe AI again

- Socio-technical benefit of GS AI: *"it makes democratic oversight easier, because concrete safety specifications can be audited and discussed by outside observers and regulators"*

- *"…if it is impossible to create world models that are sufficiently accurate to ensure that an AI system adheres to some specification, then it is presumably in general impossible to ensure that this specification is satisfied by that system."*

University of Southampton

References

- Fathabadi, Snook, Dghaym, Hoang, Alotaibi, Butler. *Systematic hierarchical analysis of requirements for critical systems,* Innovations Syst Softw Eng 2024
- Salehi Fathabadi, Asieh, Hoang, Thai Son and Butler, Michael, *SHARCS: refinement-centric hazard analysis of requirements for critical systems,* ABZ 2026
- Yeganefard, Butler, Rezazadeh, *Evaluation of a Guideline by Formal Modelling of Cruise Control System in Event-B*, NFM 2010
- Al-Shareefy, Haider, Butler, Michael and Hoang, Thai Son (2024) *AIC approach for intelligent systems requirements elicitation,* In 2023 7th International Conference on System Reliability and Safety, ICSRS 2023
- C Bogdiukiewicz, M Butler, TS Hoang, M Paxton, J Snook, X Waldron, T Wilkinson, *Formal development of policing functions for intelligent systems,* International Symposium on Software Reliability Engineering (ISSRE) 2017
- MJ Butler, JR Abrial, R Banach, *Modelling and Refining Hybrid Systems in Event-B and Rodin*, From Action Systems to Distributed Systems, 2016