UCL

Requirements Reflection ?

Anthony Finkelstein
Department of Computer Science
a.finkelstein@cs.ucl.ac.uk

Example: a context aware, adaptive, football video service

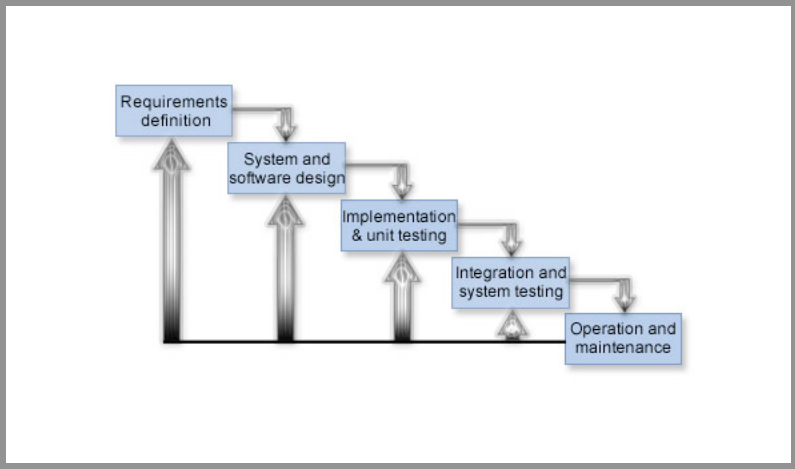


Adapts dynamically to user, bandwidth, device characteristics, location and environmental characteristics

I will use this example to motivate the talk …

Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems.

It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families.



Computational reflection is the ability of a program to observe and possibly modify its design

Typically, reflection refers to runtime or dynamic reflection, though some programming languages support compile time or static reflection.

When source code is compiled, information about the structure of the program is normally lost as lower level code is produced

If a system supports reflection, the structure is preserved as metadata.

Could we have requirements reflection? Could we dynamically observe the requirements for a software system?

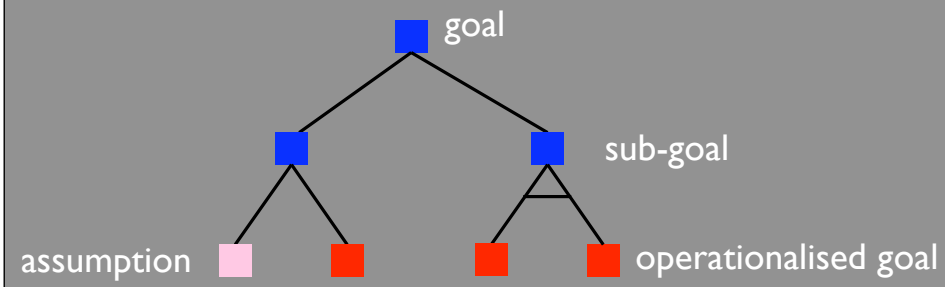
In other words can we make requirements runtime objects?

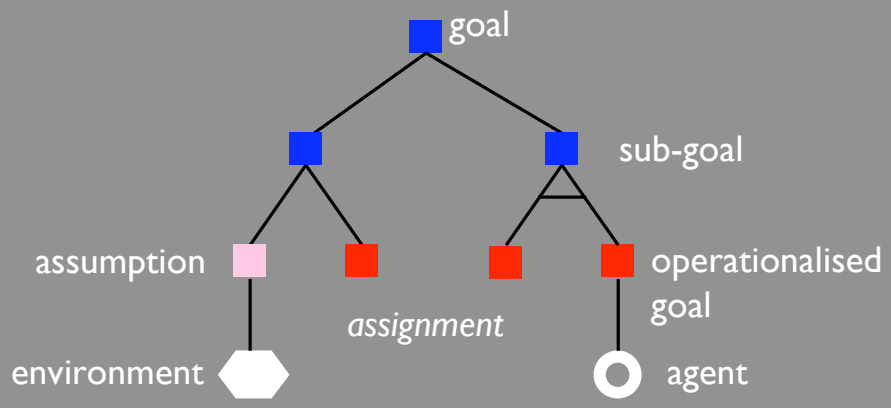
Requirements record the real-word goals for a system

We wish to satisfy (or perhaps satisfice) the goals in each context

To obtain accurate account of the standing of our favoured teams and as full a sense as possible of game highlights

KAOS, goal-oriented requirements engineering method (van Lamsweerde)





Check whether the environment will meet the assumptions

Dynamically reassign the goals to different agents

or

Move to alternative goals in the goal tree

Switch resolution, move to text only service, subtitle video …



research

- –Hard goals expressed in terms of temporal logic formulae ✓
- –Soft goals expressed in terms of metrics over predicates ✓

Generate monitors that check for environmental assumptions ✓

research

Instrument the code using Aspect Oriented Programming techniques ✓

Provide a mapping language that maps entities and relationships from goal model onto design ✓

Generate pointcuts ✓

research

Instrument the code using Aspect Oriented Programming techniques ✓

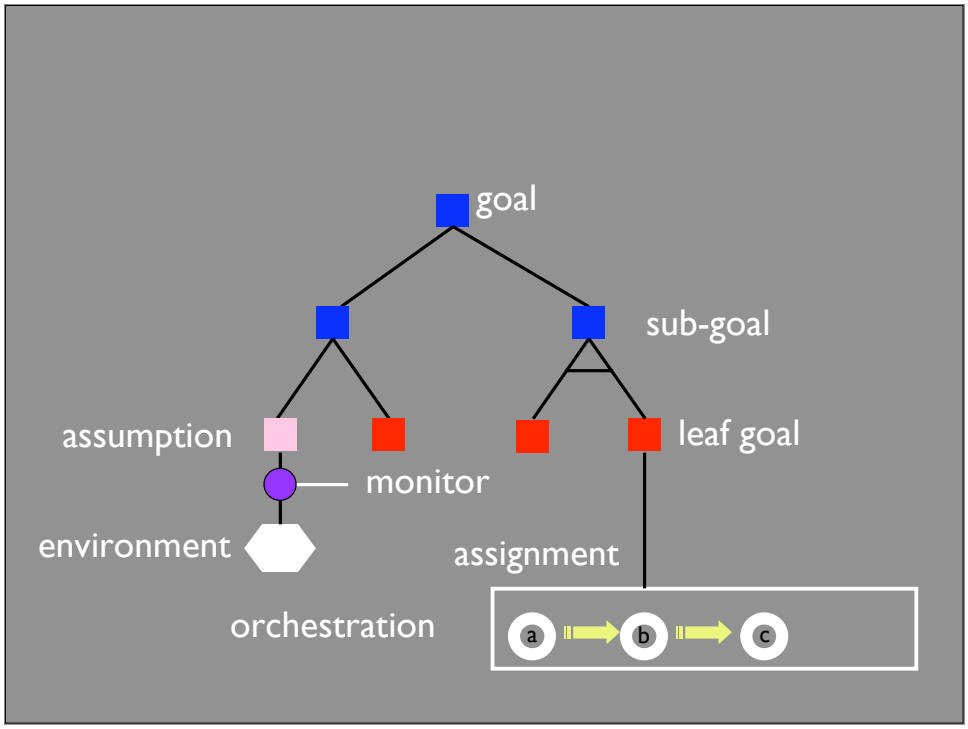
Provide a mapping language that maps entities and relationships from goal model onto design ✓

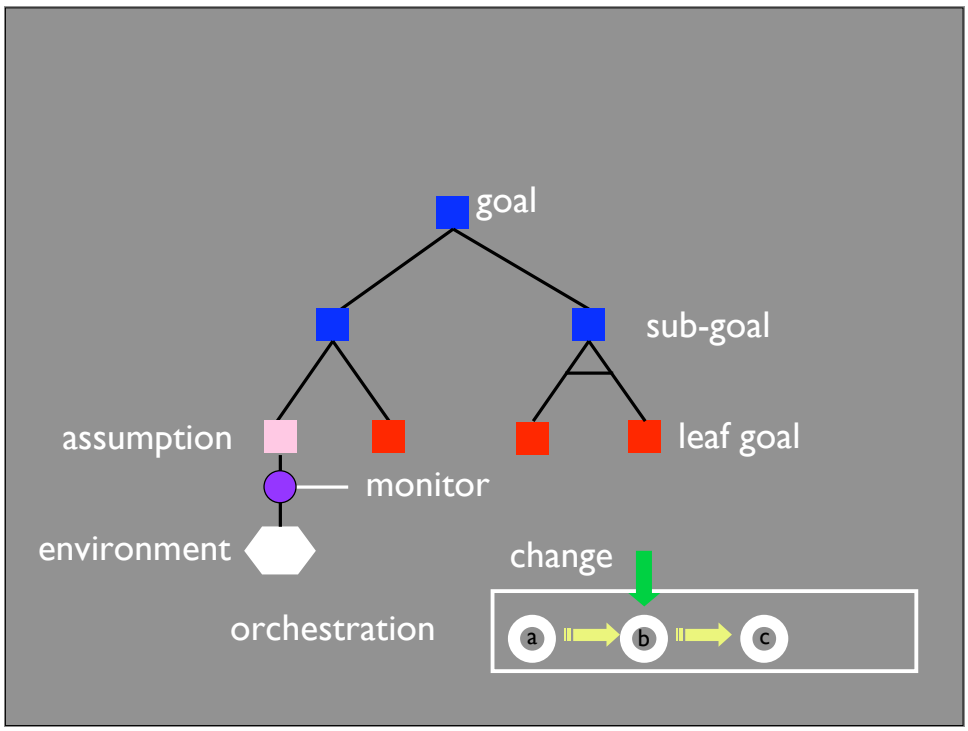
Generate pointcuts ✓

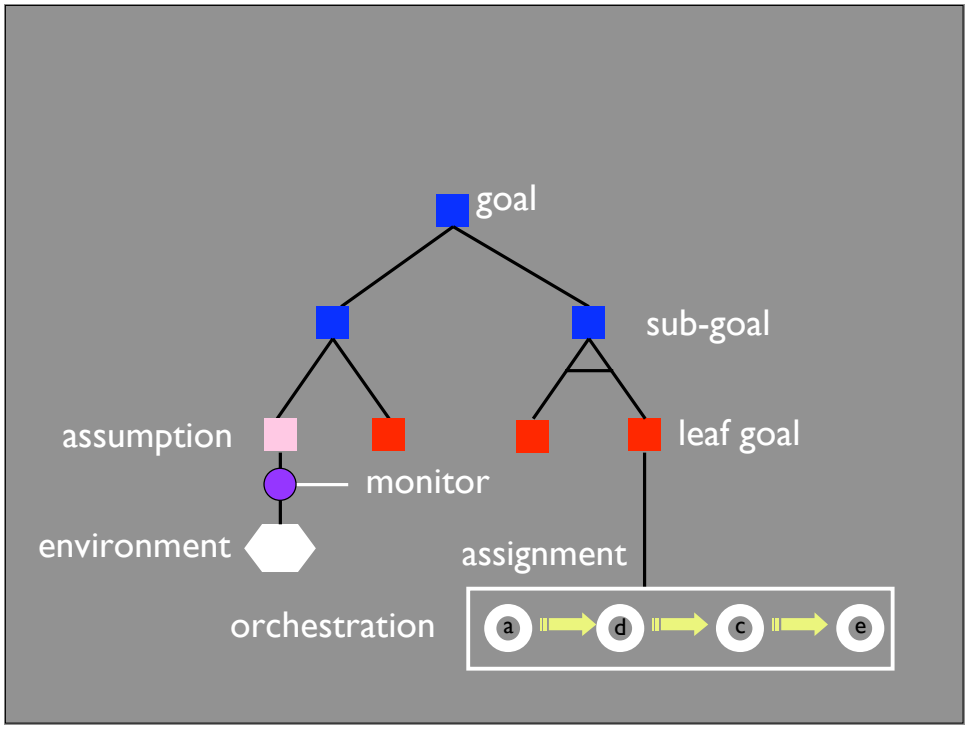
directions

Dynamic aspects for runtime adaptation

Example: BPEL + Aspects to provide dynamic service orchestration ✓


goal
sub-goal
assumption
leaf goal
monitor
environment
assignment
orchestration
a
b
c


goal
sub-goal
assumption
leaf goal
monitor
environment
change
orchestration
a
b
c



related work

–FLEA (Fickas & Feather)

–ReqMon (Robinson)

Acknowledgements:

Andy Dingwall-Smith

Carine Courbis

vision

Requirements at runtime

- As a baseline for adaptation
- To explain adaptive behaviour
- To record change and support redesign and modification

vision

Requirements at runtime

- Reuse existing resource
- To keep requirements information in sync
- To link requirements to the user configuration

Conclusion:

- Requirements engineers can contribute to the research community mix

Practical step:

- Shared testbeds and examples