Agile Requirements Engineering?

John Grundy

Professor of Software Engineering

SWINBURNE UNIVERSITY OF TECHNOLOGY

Swinburne think forward

Outline

- A bit about my experiences with RE & Agile Software Development
- Some challenges (as I see them)
- Some approaches (mine and others)
- What is still to be done (IMO)

My first experience of RE (that I can remember anyway)

- We were never taught the concept of “requirements engineering” @ UofA in mid 80s when I was a student there… (!!)
    - Or the concept of Software Engineering either
- I worked for a small software company late 80s building various ERP / GL systems
- Asked to develop Job Costing, Fleet Management systems
    - Given a data model
    - No stakeholders to gather requirements from
    - No requirements to test against
- Asked to develop Accruals system
    - Accountant as stakeholder - customer on site ☺
- What do you think happened?

My first experience of Agile (that I can remember)

- Same company (its great for war-stories to students! ☺)
- "Pair programming" – via the wheelie chair / one keyboard
- "Test-first development" – csh scripts, test DBs, batch processes
- "Stand-ups" - @ the coffee machine
  - Also my first taste of empirical methods - XX cups a day!!
- "40 hr week" – well, theoretically anyway!

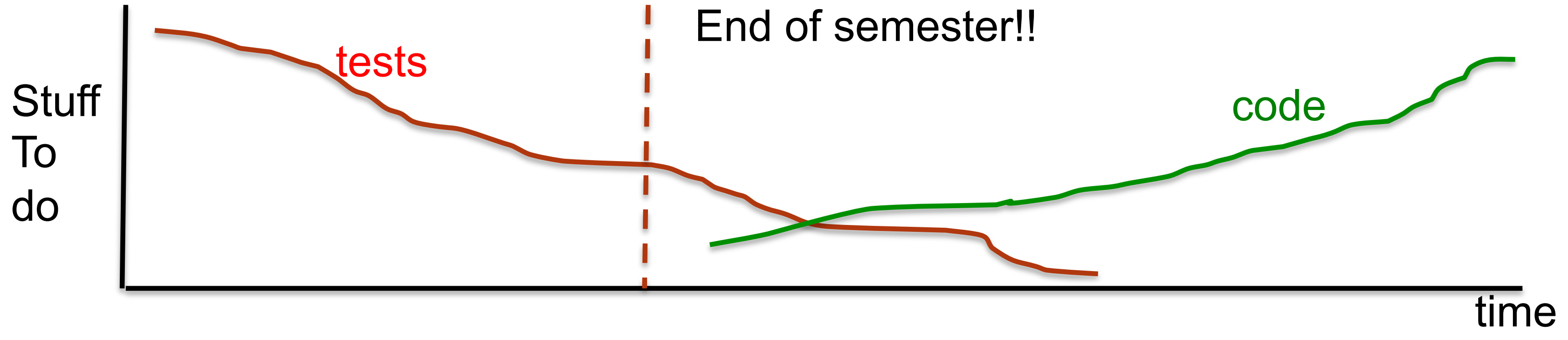
- Model-driven development – model -> 4GL/DB code
- End-user computing & MDE – bring-ups for patent application system

Where RE can go wrong if not "agile" …

- We worked on a TBG grant with another company looking at complex data-oriented systems integration
- We *thought* we understood the requirements, target end users
- We speced, rapid prototyped, tested and delivered…
- …but it turned out the target end users were totally different – and hence the carefully speced requirements totally wrong
  - "Do the right thing" vs "Do the thing right" !
  - No customer in team !!

Where Agile can go wrong if Requirements forgotten…

- Two excellent final year BE(Software) students & their capstone team project
- Personal health care planning app for mobile (this was mid-2000's!)
- Totally sold on concept of Agile and heavily adopted Test-first development approach…



[Note Phillipe Kruchen's observations on refactoring-out-of-control!!]

Automated Software Engineering & Agile/RE ?

- I like models (of software) ☺
- I like “automated” SE techniques and tools – generate code / configurations from models
- Models & RE
  - More complete & abstract the model, the better!
  - Can do various analysis of (good) models
- Models & Agile SE approaches
  - Allow rapid prototype (“spike”); “self-documenting” ☺
  - Ultimately – IMO – are far more human-centric than code – esp domain-specific (visual) lanaguages

Other (relevant) experiences…

- Teaching waterfall & agile in same unit (course)
- Working with industry teams that are anti-agile, anti-RE (sometimes both ☺)
- Trying to "invent" eXtreme Aspect-oriented Requirements Engineering (a bit more on this soon…)
- Working with software company that has standards / legislation demanding upfront requirements, very extensive requirements-based testing (ditto)
- Agile Software Architecting
- Relating Software Requirements and Architectures

Agile Software Architecting (c.f. Agile RE…)

- My forward to this new book:
  - Contrast "tayloristic" SA and Agile (specifically, XP)
  - SA perceived negatives: big design up front; rigid, intolerant of RE changes; too focused on doc vs people
  - XP perceived negatives: architectures too "emergent" esp for large systems; no doc / low doc (c.f. home loans ☺); requirements allowed to be *too* volatile
  - Various recent works on combining advantages: architecting for agile / agile SA
- Rest of this talk: can we do same for more traditional RE practices & Agile?

RE focus and Agile focus

- (Traditional) RE focus
  - Get requirements right
  - Written specification
  - Contractual doc
  - Progress to Design
  - Test to the spec
- Agile focus
  - Deliver value quickly
  - Right-size documentation
  - JIT requirements
  - Iterate, itertate, iterate
  - Test with the spec

System meets customer needs

(Paraphrasing Elke Hochmülle's Workshop on Agile RE talk)

Key (potential) benefits of (Formal) RE approaches

- Forces look before you leap (IMO – a good thing!!)
- Forces deep dialogue with stakeholders
- Formal analysis of specifications to find incomleteness, inconsistency, incorrectness early
- Enables model-based testing (or Requirements-based Testing if you prefer)
- Scales to very large scale systems of systems

Key (potential) benefits of agile software development

- Outcome-focused vs process-focused (can see the wood for the trees…) – SE is a problem-solving discipline!
- Disciplined processes e.g. XP – why I like to teach it!
- Inherently (somewhat) tolerant to requirements change
- Focus on continuous improvement (refactor, spike, replan & reprioritise etc)
- Quick delivery of value / quick get rid of no/low value

(Some of) the issues as I see them

- So why don't we always do them together??
  - Need to better leverage benefits of agile concepts / practices in traditionally non-agile domains
  - Need to better leverage benefits of RE, SE, Testing, PM practices – and modelling - in agile projects
  - How identify when to use different processes & techniques, when to blend approaches (vs all or nothing)
  - Need more human-centric models for software development
  - Need more human-centric process, tools and techniques – esp for end-user computing

Some recent work

- Some Agile Software Architecture advancements (as a comparator):
  - Tailoring SCRUM to support agile architecting
  - Continuous architecture analysis
  - Refactoring architectures

  - Mitigation of architecture deficiencies commonly found in agile projects (mostly QoS issues)
  - Driving agile practices from architecture-based RE needs (planning, priorities, spikes, refactoring, testing, …)
  - Architecture-informed agile practices

Agile and RE

- Agile Requirements Modelling (highly iterative RE))
- Collaborative RE (e.g. Wiki and other collab tools)
- Requirements on a page (conciseness is a virtue)
- EUI prototypes (I'll come back to these!)
- JIT requirements modelling
- Specification by example (scenarios, exec tests)
- Req Engineer as "liaison officer" (cost, elicit, validate)
- Agile requirements prioritisation
- Non-functional requirements reasoning in agile projects (QoS)

RE and Agile

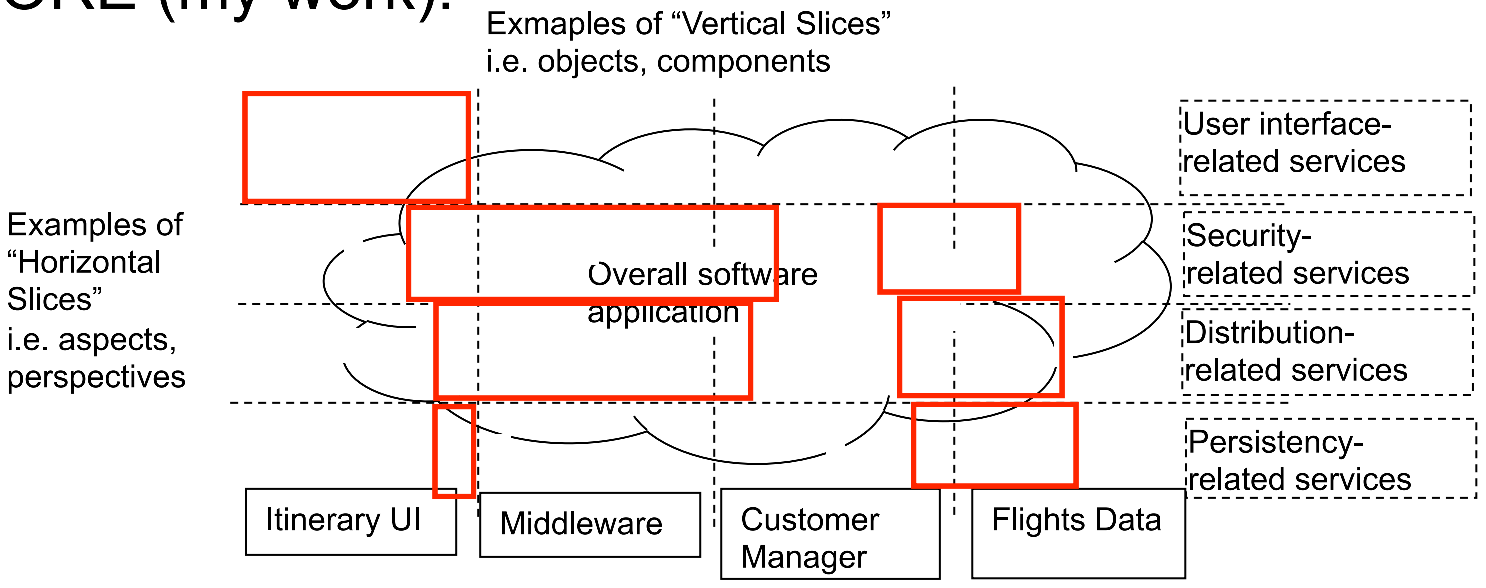
- Iterative RE (and all it implies) incl requirements refactoring
- SCRUM applied to e.g. Software Product Lines (requirements)
- Pairing for requirements analysis (c.f. PP etc)
- The Wall, story cards, planning games -> more widespread RE practices
- Team Collaboration & on-site customer concept -> more widespread RE practices
- Mock-up driven Development (another MDD ☺)
- Use cases vs user stories revisited in context of Agile RE (UC are better…!)

Some of our work in these areas

- Integrating agile practices in “heavyweight” RE approach
- Rapid app development / rapid app prototyping
- Supporting continuous architecture-based requirements analysis
- Rapid prototyping to support highly volatile requirements elicitation/refinement
- Capturing (semi)formal RE models from natural language requirements (e.g. user stories) to support upfront analysis

eXtreme AORE

- Part of Santokh Singh's PhD work
  - Ideas (1) incorporate Agile (XP in this case) concepts into "heavyweight" RE method (2) (AO) models into XP
- AORE (my work):

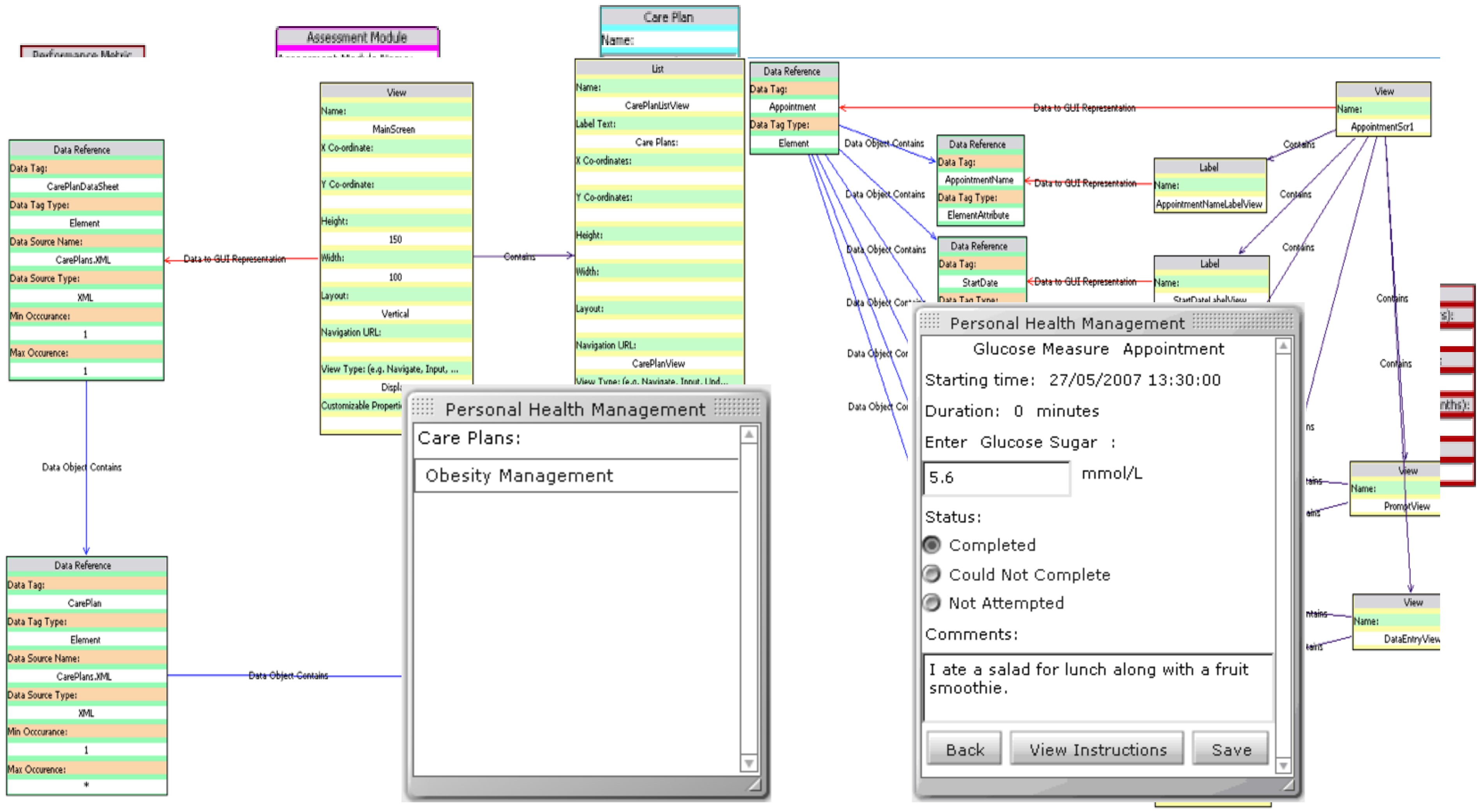

Exmaples of "Vertical Slices"
i.e. objects, components
Examples of "Horizontal Slices"
i.e. aspects, perspectives
Overall software application
User interface-related services
Security-related services
Distribution-related services
Persistency-related services
Itinerary UI
Middleware
Customer Manager
Flights Data

eXtreme AORE

- Set of XP-inspired principles incorporated into AORE
  - User stories with aspect cross-cuts identified
  - Small Releases w AO components
  - AO components and cross-cuts incl structuring, naming
  - Continuous AO-based testing (building on George Ding’s Masters work)
  - AO-based refactoring
  - AO-based PP and code/aspect ”ownership”
  - AO-based component integration
- Included an AO CVS system to support some of this
- An interesting exercise, but…

Model-based tools for exploratory & automated development

- Generating personal care apps from models - VHCPL
- Generating energy / cost / performance tests - StressCloud
- Generating mobile app prototypes from models
- Ideas:
  (1) Use high-level models to completely (or partially) generate & evolve way more rapidly
  (2) Use models & tools to do rapid refactor/re-engineer and rapid look-ahead (“spike”)



Personal Health Management

Care Plans:

Obesity Management

Personal Health Management

Glucose Measure Appointment

Starting time: 27/05/2007 13:30:00

Duration: 0 minutes

Enter Glucose Sugar : 5.6 mmol/L

Status:

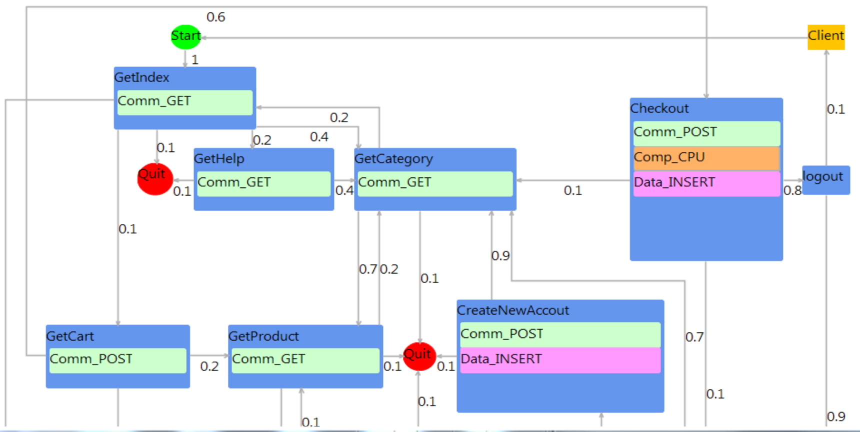
[✓ Completed]
Could Not Complete
Not Attempted

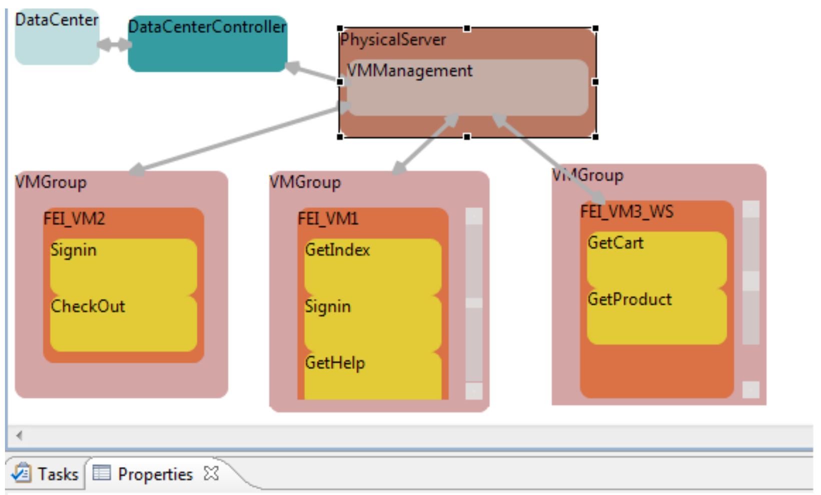
Comments:

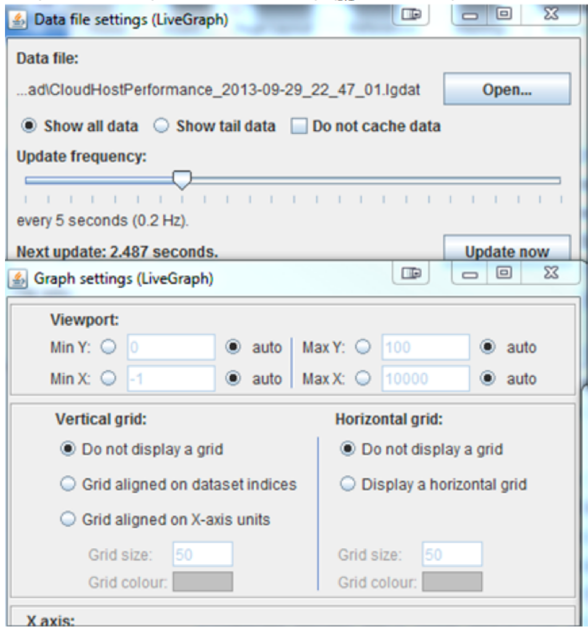
I ate a salad for lunch along with a fruit smoothie.

Back | View Instructions | Save

Generate Performance etc tests






Data file settings (LiveGraph)

Data file:

...ad\CloudHostPerformance_2013-09-29_22_47_01.lgdat Open...

[✓ Show all data] Show tail data Do not cache data

Update frequency:

every 5 seconds (0.2 Hz).

Next update: 2.487 seconds. Update now

Graph settings (LiveGraph)

Viewport:

Min Y: 0 [✓ auto] Max Y: 100 [✓ auto]

Min X: -1 [✓ auto] Max X: 10000 [✓ auto]

Vertical grid:

[✓ Do not display a grid]

Grid aligned on dataset indices

Grid aligned on X-axis units

Grid size: 50

Grid colour:

Horizontal grid:

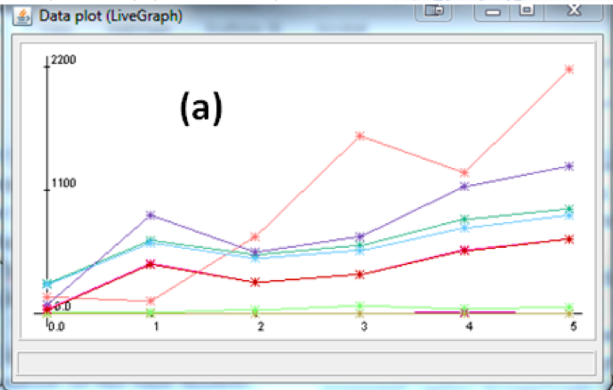
[✓ Do not display a grid]

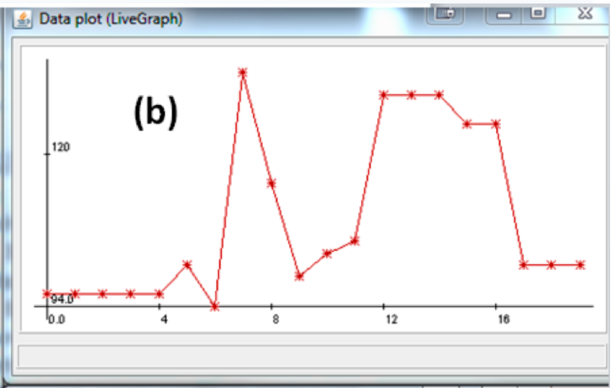
Display a horizontal grid

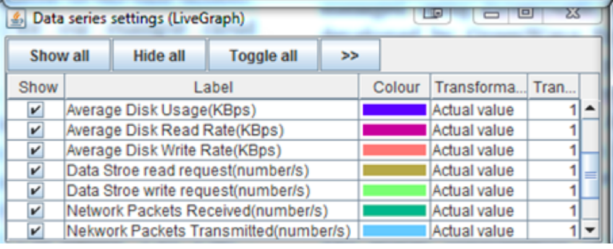
Grid size: 50

Grid colour:

X axis:

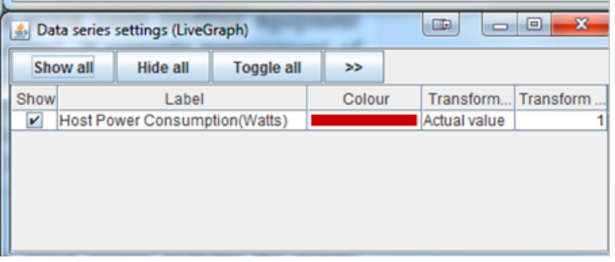





Data series settings (LiveGraph)

Show all | Hide all | Toggle all | >>

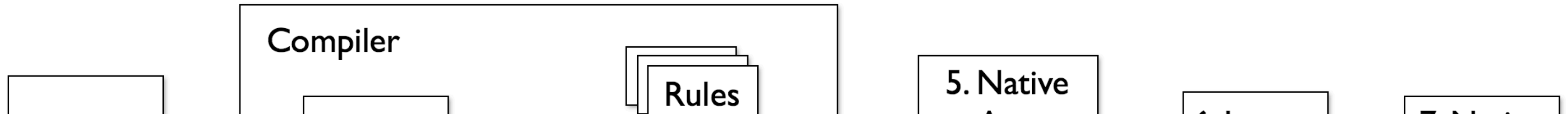
Show	Label	Colour	Transforma...	Tran...
✓	Average Disk Usage(KBps)	🟪	Actual value	1
✓	Average Disk Read Rate(KBps)	🟪	Actual value	1
✓	Average Disk Write Rate(KBps)	🟥	Actual value	1
✓	Data Stroe read request(number/s)	🟨	Actual value	1
✓	Data Stroe write request(number/s)	🟩	Actual value	1
✓	Network Packets Received(number/s)	🟩	Actual value	1
✓	Nekwork Packets Transmitted(number/s)	🟦	Actual value	1

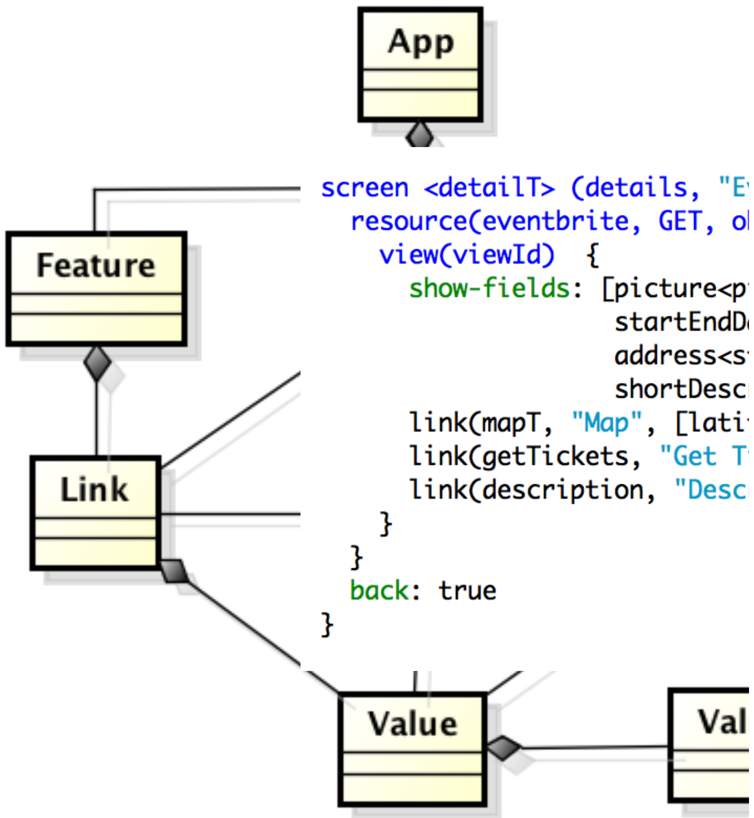

Data series settings (LiveGraph)

Show all | Hide all | Toggle all | >>

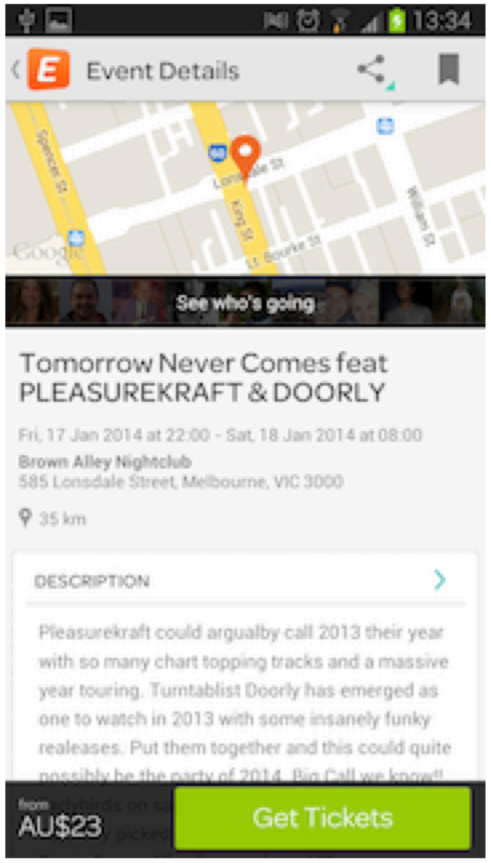
Show	Label	Colour	Transform...	Transform...
✓	Host Power Consumption(Watts)	🟥	Actual value	1

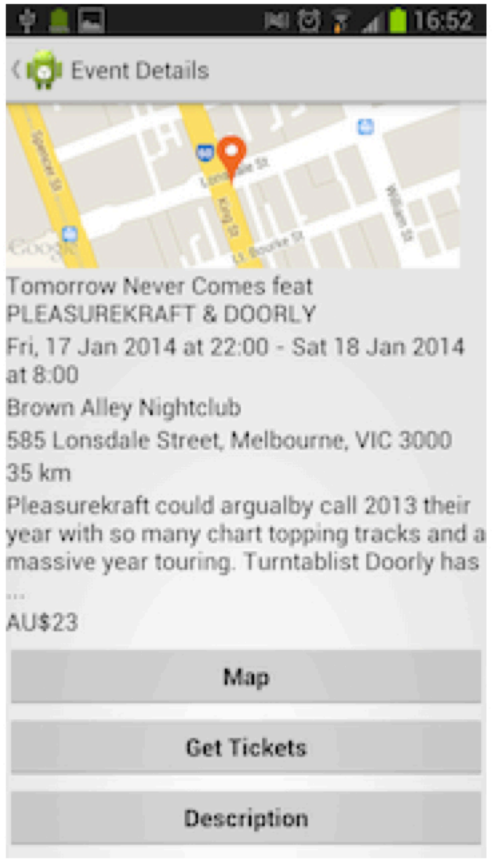
Generate mobile app prototypes – RAD makes a come-back!





```
screen <detailT> (details, "E
  resource(eventbrite, GET, o
    view(viewId)  {
      show-fields: [picture<p
                    startEndD
                    address<s
                    shortDesc
      link(mapT, "Map", [lati
      link(getTickets, "Get T
      link(description, "Desc
    }
  }
  back: true
}
```

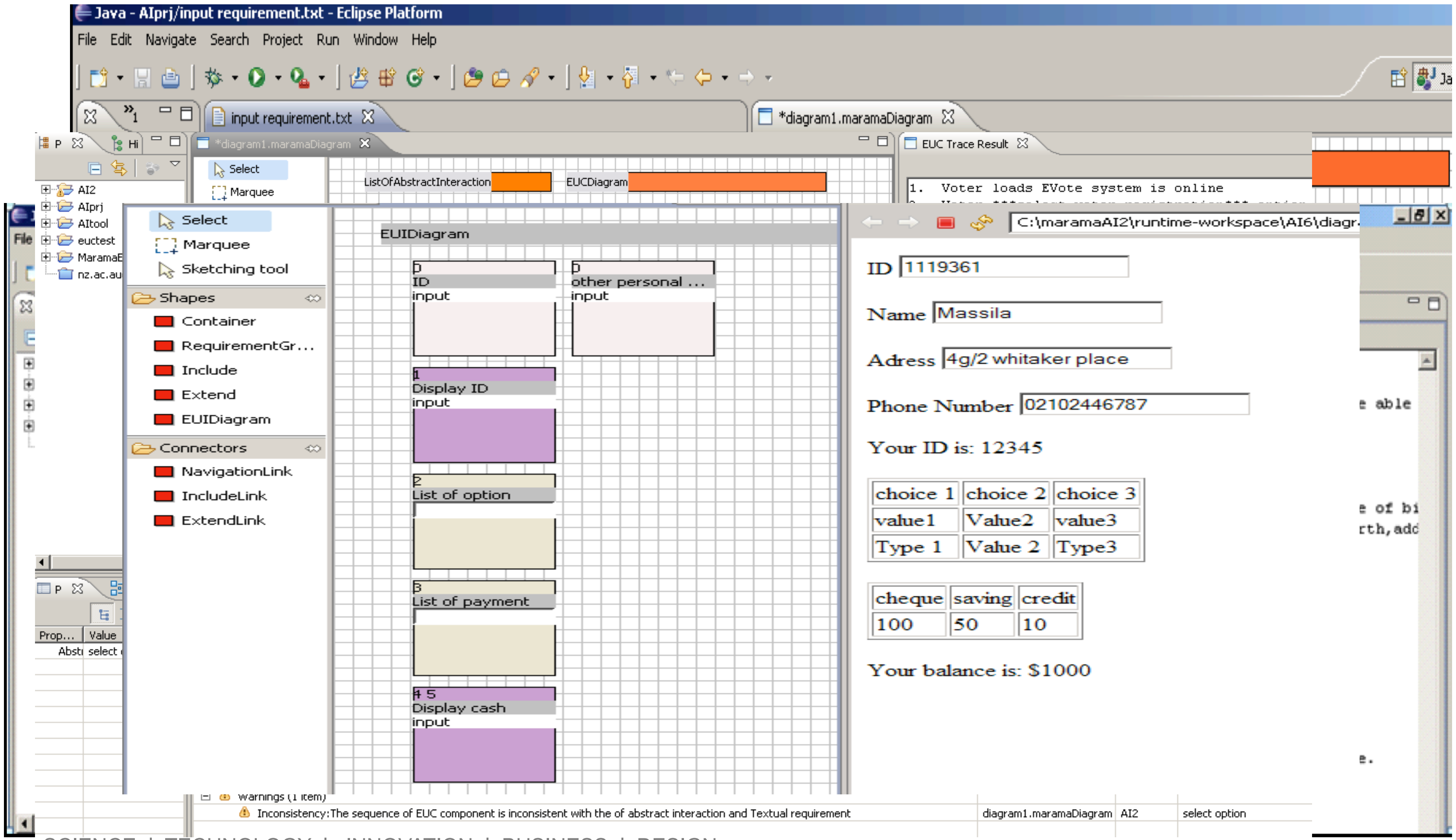

13:34
Event Details
See who's going
Tomorrow Never Comes feat PLEASUREKRAFT & DOORLY
Fri, 17 Jan 2014 at 22:00 - Sat, 18 Jan 2014 at 08:00
Brown Alley Nightclub
585 Lonsdale Street, Melbourne, VIC 3000
35 km
DESCRIPTION
Pleasurekraft could argualby call 2013 their year with so many chart topping tracks and a massive year touring. Turntablist Doorly has emerged as one to watch in 2013 with some insanely funky realeases. Put them together and this could quite possibly be the party of 2014. Big Call we know!!
from AU$23
Get Tickets


16:52
Event Details
Tomorrow Never Comes feat PLEASUREKRAFT & DOORLY
Fri, 17 Jan 2014 at 22:00 - Sat 18 Jan 2014 at 8:00
Brown Alley Nightclub
585 Lonsdale Street, Melbourne, VIC 3000
35 km
Pleasurekraft could argualby call 2013 their year with so many chart topping tracks and a massive year touring. Turntablist Doorly has
...
AU$23
Map
Get Tickets
Description

MEReq, GUITAR, Integration Mock-ups

- Enable rapid UI prototying (MEReq) from EUCs to support dialogue between RE and stakeholders
- Extract requirements models from NL text (MEReq, GUITAR) and apply pattern and ontoloty based analysis
- “Executable” mock-ups of system integration points to capture flow of complex system interactions
- Ideas are to (1) improve stakeholder understanding of (implications of) captured requirements; (2) early phase check requirements 3Cs; (3) deeper engagement with requirements by stakeholders

MEReq



```
1.  Voter loads EVote system is online
```

ID 1119361

Name Massila

Adress 4g/2 whitaker place

Phone Number 02102446787

Your ID is: 12345

choice 1	choice 2	choice 3
value1	Value2	value3
Type 1	Value 2	Type3

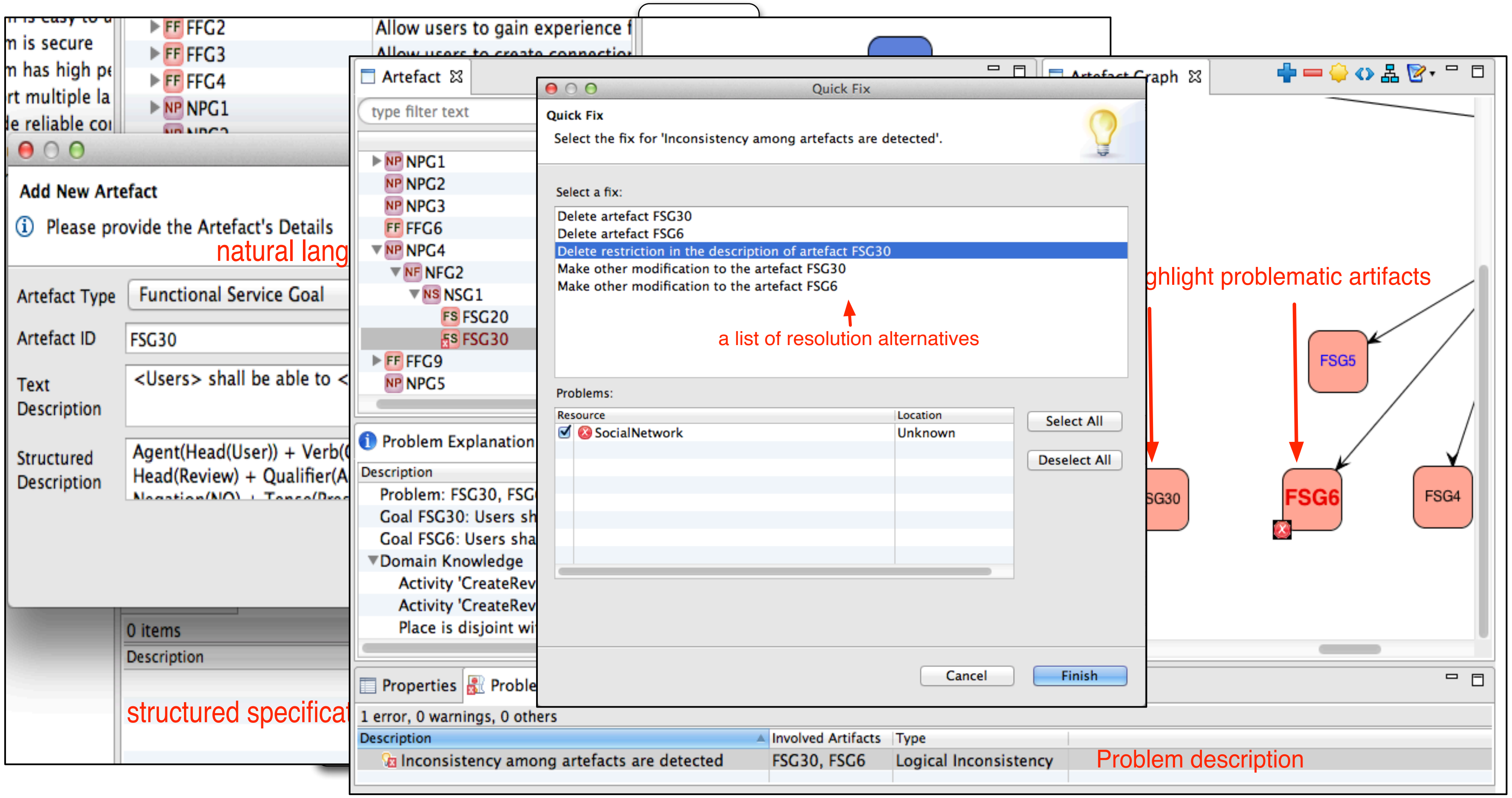
cheque	saving	credit
100	50	10

Your balance is: $1000

Warnings (1 item)

Inconsistency:The sequence of EUC component is inconsistent with the of abstract interaction and Textual requirement | diagram1.maramaDiagram | AI2 | select option

GUITAR analysis


natural lang
structured specificat
a list of resolution alternatives
ghlight problematic artifacts
Problem description

Add New Artefact

Please provide the Artefact's Details

Artefact Type	Functional Service Goal
Artefact ID	FSG30
Text Description	<Users> shall be able to <
Structured Description	Agent(Head(User)) + Verb((Head(Review) + Qualifier(A

Quick Fix

Select the fix for 'Inconsistency among artefacts are detected'.

Select a fix:

- Delete artefact FSG30
- Delete artefact FSG6
- Delete restriction in the description of artefact FSG30
- Make other modification to the artefact FSG30
- Make other modification to the artefact FSG6

Problems:

Resource	Location
✓ SocialNetwork	Unknown

Select All | Deselect All | Cancel | Finish

Problem Explanation

Description
- Problem: FSG30, FSG
- Goal FSG30: Users sh
- Goal FSG6: Users sha
- Domain Knowledge
  - Activity 'CreateRev
  - Activity 'CreateRev
  - Place is disjoint wi

1 error, 0 warnings, 0 others

Description	Involved Artifacts	Type
Inconsistency among artefacts are detected	FSG30, FSG6	Logical Inconsistency

System Integration mock-ups

- Rapid prototype system integration mockups
- Capture main integration points, flow of control
- Use video to capture thinking / rationale
- Use web-based / Tablet-based mock-up of system in / out / sequencing
- Evidence of much deeper engagement with requirements than previous user story / use case / UI mock-ups…

A brief aside… Personality and Agile practices / RE / Testing

- We have also been studying
  - Impact of personality of pair programmers (in teaching setting for introductory / intermediate programming units)
  - Impact of pairing on requirements engineering practices (both industry practitioners and students)
  - Impact of personality on software testing competency (industry practitioners and students)
- (Some aspects of) Personality of the developer does impact (in someways) RE / PP / testing compentencies
- How do we leverage this knowledge??? i.e. the human factors

Outstanding issues

- Deploying formal analysis early – need detailed specs
- Scaling – see Philippe Krutchen's lovely examples
- System of systems – need to integrate into complex architectures
- Security critical, safety critical domains / issues
- How to cost projects, manage costs
  - What models need & how get them?

How do we educate “Agile Requirements Engineers” ?

- Need deep domain concept understanding / skills to acquire: act as/with stakeholders
- Need good models to express requirements for whole team (stakeholders, developers, BAs…)
- Executable models a la FitNesse, MBT techniques
- Rapid prototypes e.g. apps, processes v useful for dialogue with stakeholders
- Rapid idea / architecture analysis ; what-if-ing
- Team dynamics – customer, RE, developer, … ?

A challenging example domain

- Working with a company that has:
  - Legislated need for very detailed requirements models
  - Legislated need for model-based testing i.e. test against requirements with no knowledge of arch / design / impl
  - Systems of systems – literally hundreds of systems – to fit together
  - Systems engineers averse to highly mathematical models
  - Company / regulators averse to “agile” concepts
  - Models can be leveraged to generate MBT, code, integration frameworks etc

Conclusions

- Agile software development and (formal) Requirements Engineering have advantages and limitations
- Their strengths can mitigate each others weaknesses
- Models (and good tool support!) are the key (IMO):
  - Far more human-centric than code
  - Domain abstractions can be much better leveraged
  - Model-based tools assist in verifying, validating, generating tests, refactoring, assessing quickly (spikes), …
  - Need to work with informal, semi-formal, formal models
  - Some domains are still… very challenging to apply both RE and Agile!

SWIN BUR NE

SWINBURNE UNIVERSITY OF TECHNOLOGY

Comments / Questions

Thanks for funding support to:

Ministry of Business, Innovation & Employment

FRST SER SPPI

Australian Government
Australian Research Council

DP110101340,
LP130100201

STUDY MELBOURNE VICTORIA AUSTRALIA

VIRS

GOVERNMENT OF MALAYSIA

Malaysian Ministry of Education

Swinburne
think forward