MONASH University

MONASH
INFORMATION
TECHNOLOGY

Human-centric (Issues in) Software Engineering

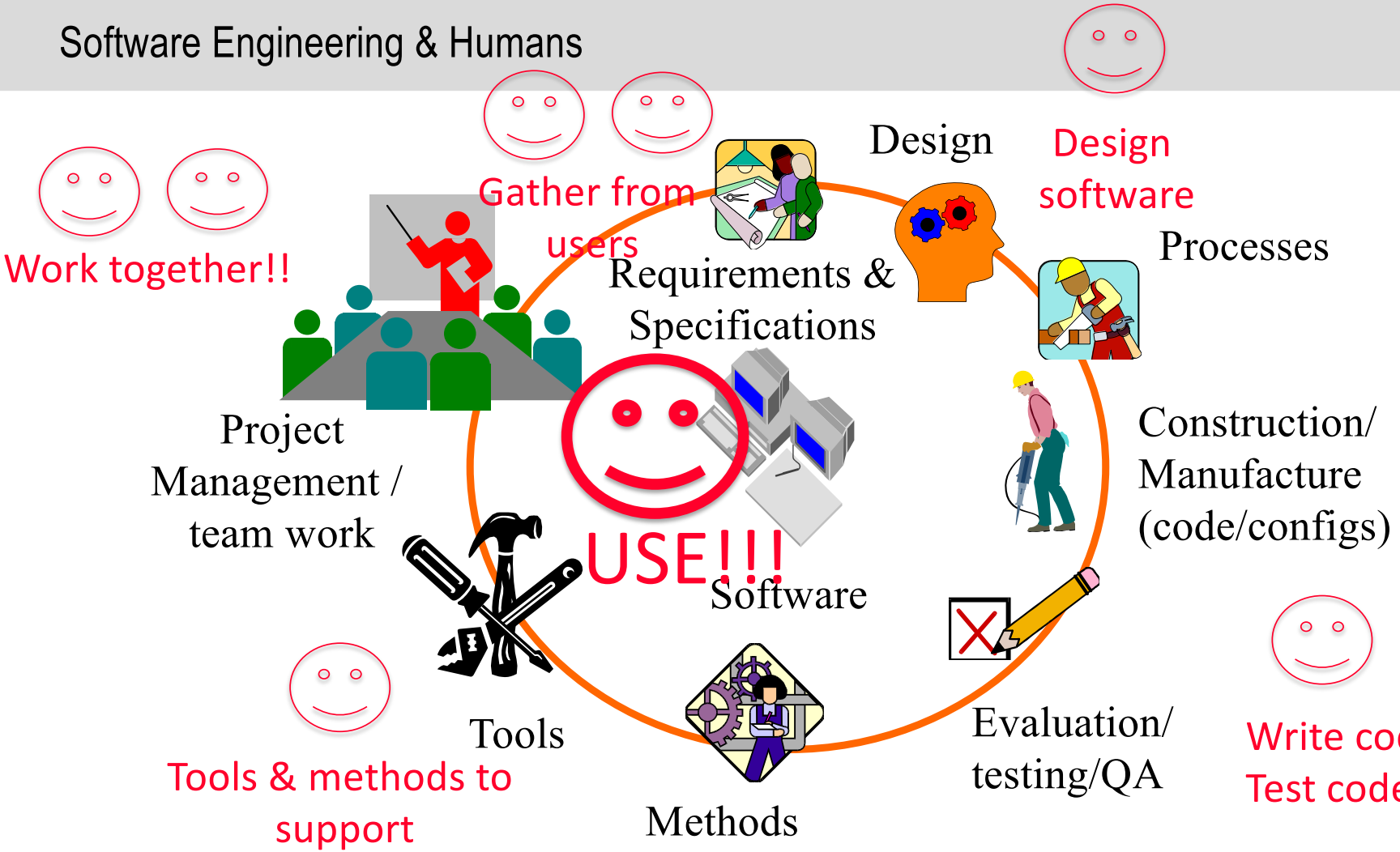
John Grundy

Australian Laureate Fellow and Professor of Software Engineering

GROUP OF EIGHT AUSTRALIA

Outline

- Software Engineering & humans
- Examples from our work
  - Human-centric, domain-specific visual models for non-technical experts to specify and generate systems
  - Multi-lingual requirements engineering
  - Incorporating end user emotions into requirements engineering
  - Personality impact on aspects of software development
  - Reporting usability defects
- Challenges, issues and future directions

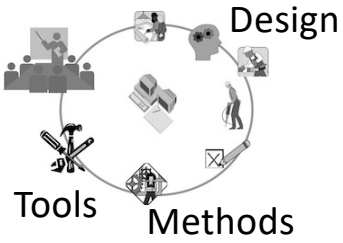

Work together!!
Gather from users
Design
Design software
Processes
Requirements & Specifications
Project Management / team work
USE!!!
Software
Construction/ Manufacture (code/configs)
Tools
Tools & methods to support
Methods
Evaluation/ testing/QA
Write code, Test code

Problems if we don't include human perspective…

- Gender bias – UIs, seat belts, health app
- Ethnic bias – over-recommend minorities for search, don't recognize faces
- Culture bias – inappropriate words, phrases, colours, icons, workflow
- Language bias – over-technical, wrong dialect, impersonal
- Age bias – too complex, too simple, inappropriate words, symbols, workflow
- Physical challenge bias – gesture, sound, sight, voice inappropriate
- Cognitive challenge bias – raise anxiety, poor fit to mental model
- Enjoyment bias – boring, unengaging, distracting
- Emotional bias – stressful, anxiety-inducing, frightening
- Personality bias – workflow, lack of engagement, disconnected

All Can Apply to TEAM and USERS!!!

Human-centric, domain-specific visual models



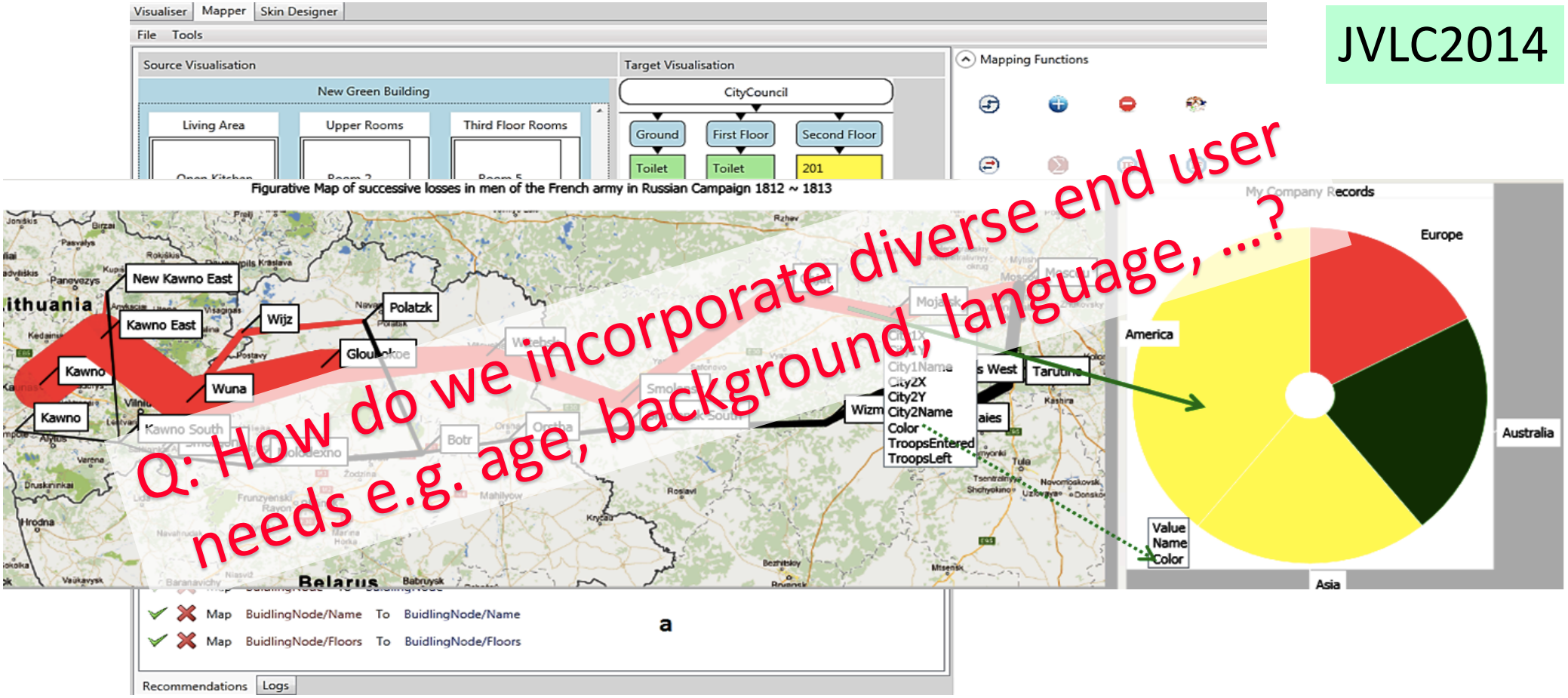
- Idea: complex models hard to work with for developers
  - And non-develpers!!
- Represent using more ”human-centric” way – visual metaphors, visual constructs – “like what sketch on a napkin in a café…” ☺
- (very) Large body of work on this (200+ papers):
  - Platforms – MViews, JViews, Pounamu, Marama, Horus, …
  - Software Engineering uses – Design tool generators, software architecture, performance engineering, user interfaces, requirements, testing, software visualisations, traceability, …
  - “End-user” Application modelling and generation – Statistical Design Language, Report Generation Language, Mobile Health App generation, Business processes, Music, Games, Visual Wikis, …

Example #1: Data integration

- Scenario: complex XML or EDI message format; want to translate into a different format; then process e.g. data wrangling, harmonization ☺
- Traditionally: write QVT/ATL/XSLT/code to do
- Alternative: model transformation visually and generate these transformation implementations
- Meta-model = source/target and mappings
- Visual models might include forms, trees, concrete data visualisations
- Model-driven Engineering = generate XSLT, ATL, Code (C++, Java),…
- Done various with Orion Health Ltd, XSOL Ltd, NICTA/Data61, …

CONVErT – by-example based data mapping/integration/visualisation

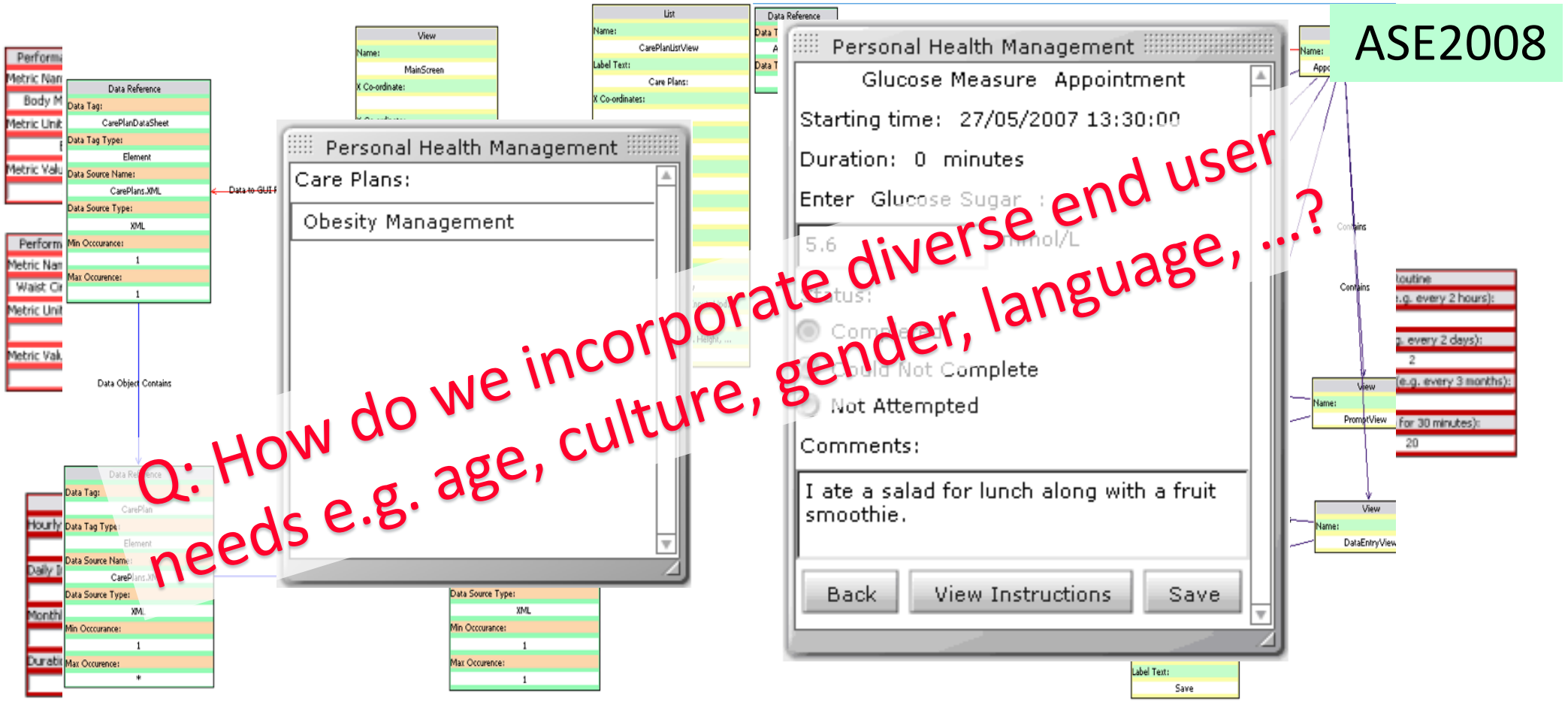
JVLC2014



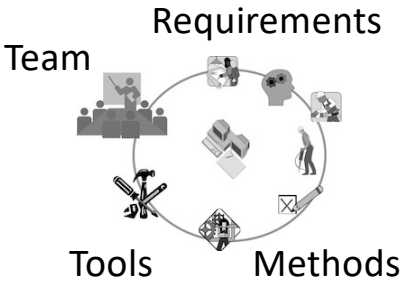
Example #2: Mobile Health app generation

- Scenario: want to model, generate range of eHealth apps
- Mobile phone-based personal health care planning applications
- Two meta-models with associated DVSLs: Visual Health Care Planning Language, Visual Care Application Model
- Model generic care plan with a visual DSVL tool
- Configure generic care plan for individual
- Model mobile app UI for individual from tailored care plan with a visual DSVL tool
- Generate Flash, Windows Mobile, iPhone app code

VHCPL


ASE2008
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
Not Attempted
Comments:
I ate a salad for lunch along with a fruit smoothie.
Back
View Instructions
Save
Data Reference
Data Tag:
CarePlanDataSheet
Data Tag Type:
Element
Data Source Name:
CarePlans.XML
Data Source Type:
XML
Min Occurance:
1
Max Occurence:
1
Data Object Contains
Contains
Q: How do we incorporate diverse end user needs e.g. age, culture, gender, language, …?

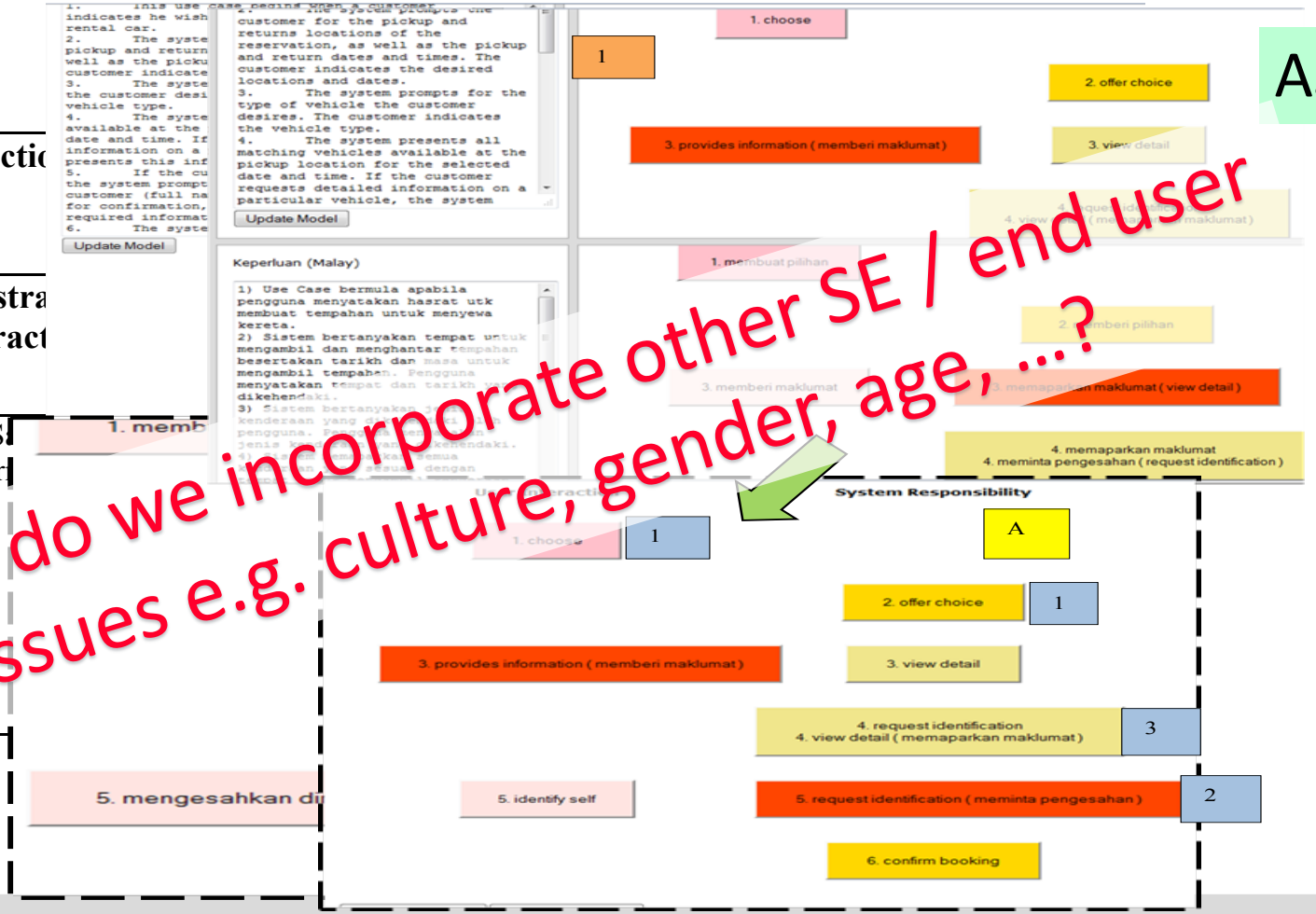
Multi-lingual Requirements Engineering



- Software developed by teams
- Teams may be diverse in many ways
  - Location
  - Language
  - Gender
  - Culture
  - Organization
- Explored one aspect in Malaysian context with multi-lingual teams (also have multi-cultural aspect)
- Added multi-lingual support to Essential use case-based requirements tool

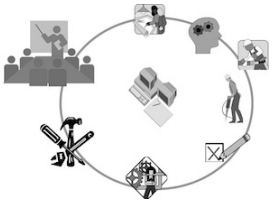
ASEJ2017

English Essential Interaction Patterns Library	
Essential Interaction	**Abstract Interaction**
1. Save record 2. Save information 3. Save data	Save information



Q: How do we incorporate other SE / end user issues e.g. culture, gender, age, …?

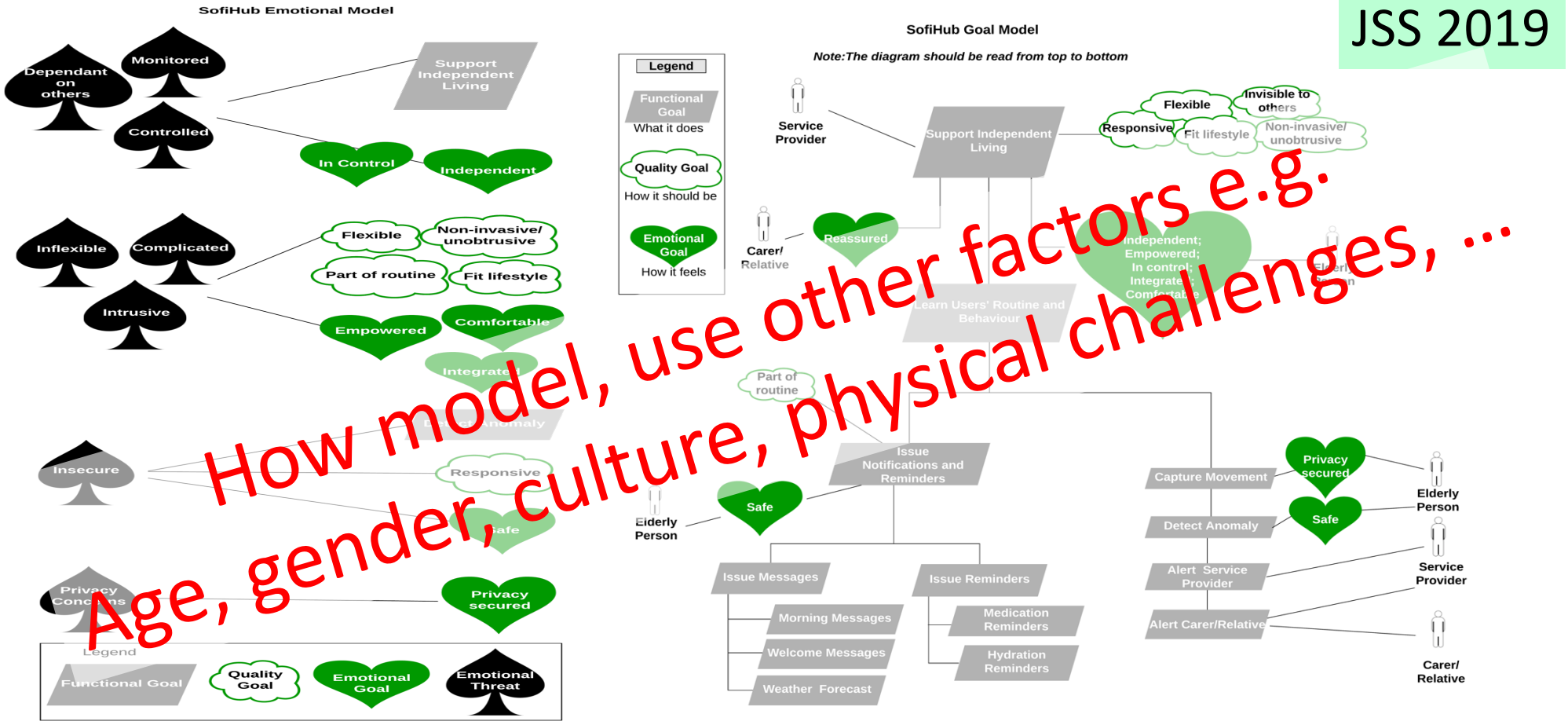
Incorporating end user emotions into software requirements engineering



- People use software
- Software is designed to help people perform tasks, solve problems
- But – people react to software / tasks / situations in various ways
- One (under-researched) way is emotional reactions to software usage
- Incorporating emotions / emotional reactions into software requirements, design, evaluation
- Applying to eHealth systems

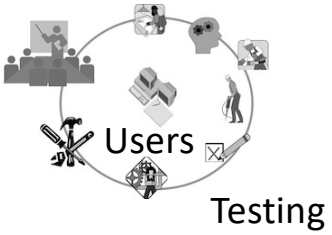
Example: requirements for the Smart Home

JSS 2019



How model, use other factors e.g. Age, gender, culture, physical challenges, …

Reporting usability defects



- Software typically has a bunch of “defects”
- Functional and non-functional
- One under-researched non-functional area are usability defects
  - Problems with how users interact with the software
- How do we currently find, report, fix these?
- How can we improve the reporting?
- Better understand current reporting needs: survey, repository mining, observation
- New usability defect taxonomy to better characterise usability defects
- New usability defect reporting tool

Usability Defect Taxonomy & Reporting

TSE2017

How better report human-centric defects e.g. Age, gender, culture, physical challenges, …

Challenges ; Outstanding issues

- Often software engineers don't understand / appreciate / not trained in human aspects of SE
- Neither it seems do MBIE or ARC (NZ and Oz grant bodies) Assessors …! ☹
- Designing and conducting experiments is hard, time-consuming
- Often need access to practitioners ; convincing them/their bosses a challenge
- Many issues not yet well explored, but increasing interest in SE community
- I find them more challenging – but also in many ways more interesting – projects than the purely technical ones I do
- Recruiting (very good) students / post-docs to work on can be hard, but I've been pretty lucky to date…
- IMO – good research in these areas can make a major difference to practice

Future work

- Adding Emotions, accessibility, personalilty etc -> UML etc models
- Capturing, using further human-centric issues: values, emotions, usability, accessibility, culture, language, gender, age, … & evaluating software for these
- Incorporating multi-lingual, multi-cultural aspects into requirements, design
- Deep learning + design critics + PM
- Agile SE Team Climate Inventory & applying in practice
- Personality of requirements engineers, software architects, project managers
- DSVLs for Big Data applications, end user config incl security
- Better principles, tools for human-centric DSVL design & evaluation

Summary

- Human aspects of Software Engineering are fascinating!!
- There is lots of scope for work here
- Can apply other discipline approaches, knowledge – Information Systems, Social Sciences, etc
- Ultimately humans PRODUCE software and humans USE software
- Incorporating human perspectives critical to improve software and its production

Questions…

References

- Grundy, J.C, Hosking, J.G., Amor, R., Mugridge, W.B., Li, M. Domain-specific visual languages for specifying and generating data mapping system, Journal of Visual Languages and Computing, vol. 15, no. 3-4, June-August 2004, Elsevier, pp 243-263
- Avazpour, I., Grundy, J.C., Grunske, L. Specifying Model Transformations by Direct Manipulation using Concrete Visual Notations and Interactive Recommendations, Journal of Visual Languages and Computing, Volume 28, June 2015, Elsevier,pp 195–211.
- Abizer Khambati, John Grundy, John Hosking, and Jim Warren, Model-driven Development of Mobile Personal Health Care Applications, In Proceedings of the 2008 IEEE/ACM International Conference on Automated Software Engineering, L'Aquilla, Italy, 15-19 September 2008, IEEE CS Press
- Kamalrudin, M., Grundy, J.C., Hosking, J.G., MaramaAIC: Tool Support for Consistency Management and Validation of Requirements, Automated Software Engineering, Springer, 2017, vol 24, no 1, pp. 1-45
- Sallah, N., Mendes, E., Grundy, J.C. Investigating the effects of personality traits on pair programming in a higher education setting through a family of experiments, Empirical Software Engineering, vol. 19, no. 3, Springer, 2014, pp. 714-752.
- Kanij, T., Merkel, R., Grundy, J.C. Performance Appraisal of Software Testers, Information and Software Technology, Elsevier, vol. 56, no. 5, May 2014, Pages 495–505
- Yusop, N.S.M., Grundy, J.C., Vasa, R. Reporting Usability Defects: A Systematic Literature Review, IEEE Transactions on Software Engineering, vol. 43, no. 9, 2017, pp. 848-867.
- Ali, N.M., Hosking, J.G., Grundy, J.C., A Taxonomy and Mapping of Computer-based Critiquing Tools, IEEE Transactions on Software Engineering, vol. 39, no. 11, November 2013.
- Grundy, J.C. Abdelrazek, M., Kissoon, M., Vision: Improved development of mobile eHealth applications, IEEE/ACM International Conference on Mobile Software Engineering and Systems (MobileSoft 2018), 27-28 May 2018, Gothenberg, Sweden, ACM Press.
- Salleh, N., Hoda, R., Su, M.T., Kanij, T. and Grundy, J.C. Recruitment, Engagement and Feedback in Industrial Empirical Software Engineering Studies, to appear in Information and Software Technology, Elsevier. –
- Soomro, A.B., Salleh, N., Mendes, E., Grundy, J.C., Burch, G., Nordin, A., The Effect of Software Engineers' Personality traits on Team Climate and Performance: a Systematic Literature Review, Information and Software Technology, vol 73, Elsevier, pp 52-65.