Requirements Engineering in the Days of Continuous Delivery

How can AI help?

Anna Perini
Fondazione Bruno Kessler, Italy
perini@fbk.eu

FBK, Povo - Nov. 7, 2018
SUPERSEDE, ended successfully!

The SUPERSEDE project  www.supersede.eu

- **SU**porting evolution and adaptation of **PER**sonalized Software by Exploiting contextual **D**ata and **E**nd-user feedback

- An European funded project (**HORIZON 2020** framework, RIA)
  - Call: H2020-ICT-2014-1 (Tools and methods for Software Development)
  - Started May 1, 2015 – ended April 30, 2018
  - Consortium: 4 Academic/Research and 4 Industrial partners

... time to reflect on results with a broader perspective
Requirements Engineering

- RE aims at ensuring “design for purpose”
- Software requirements capture the needs of the users of such software (stakeholders)

Are we building the right system?
How do we perform RE?

- RE provides methods and techniques for:
  - eliciting, analysing, prioritizing, managing changes and evolution of software requirements

- RE is a **key** process, which is (usually) performed before starting to design and develop a software system

- RE tasks are **decision**-intensive
Observed trend in software development

- **move fast**
  - E.g. Facebook’s Android, from a release every 8 weeks to a release every week\(^1\)
    - i.e. adopting incremental, continuous delivery of a small number of functionalities

- **Agile and continuous delivery**
  - NaPiRE survey (196 companies):
    - **47% Agile, 30% mixed Agile/ «plan-driven»**\(^2\)

---
\(^1\) T. Savor Et al., FSE ’16
\(^2\) Méndez Fernández et al. NaPiRE - Naming the pain in requirements engineering - 2014-15 (EU, NA, SA) - EMSE17
Continuous delivery ... challenging RE

- Requirements artefacts (feature, story, test cases)
- Traceability
- ....
- Continuous **assessment of** the effect of a new software release
- Collecting and analysing **usage data** and user feedback at support of informed decisions
- Effective (multi-criteria) **requirements prioritization**
Outline

✓ Setting the scene
  • Data-driven Requirements Engineering
    - The SUPERSEDE way
  • Conclusion:
    - Open challenges
    - AI & RE
• Special thanks
Data-driven requirements engineering
Data-driven RE definition

• Data-driven RE rests on *data generated at execution/usage time* that are *automatically collected*, and *analysed* with the purpose of supporting *RE decisions*

• Example:
  Which requirements should be implemented in the next software release?

* Maalej et al., IEEE Software, 2016
Data-driven RE ingredients

1. Data
2. Techniques for data processing & analysis
3. Decision-making needs
Data-driven RE Ingredients

1. Data
Software at use

SMART CITY INFORMATION PLATFORM

- Online data
  - Smart City Information API provision and consumption
  - Ecosystem for Smart City Information exchange

Online User feedback

- Social media
- Usage monitoring

- Online data sources
  - sensors embedded in the environment / telemetry
- explicit and implicit users’ feedback

Explicit user feedback

• Types
  • Linguistic: natural-language text, audio messages
  • Non-linguistic: images, emoticons, and star ratings
  • Multi-modal feedback
    • E.g. App review: timestamp showing when the review was created, a user rating, short textual comment

• Gathering Tool & Platforms
  • Feedback-gathering tools (e.g. SUPERSEDE-FG tool)\(^3\)
  • App stores
  • Forums
  • Issue tracking systems
  • Social media platforms, e.g. Twitter

---

\(^3\) SEnerCon iESA application [www.supersede.eu](http://www.supersede.eu); Groen et al. IEEE Software 2017
Implicit user feedback

- **Types**
  - User’s behavioural data
    - click stream and navigation paths in web-based applications
- **Monitoring tools**
  - ReqMon (W.N. Robinson REJ 2006)
    - real-time feedback on requirements satisfaction
  - SUPERSEDE integrated monitors (see picture)

*Note*: can be combined with other software system run-time monitored data

---

*Oriol et al. RE18*
Data-driven RE Ingredients

2. data processing & analysis
Analysis approaches - Explicit user feedback

- NLP techniques for preprocessing
- Computational linguistic techniques to
  - Identify patterns (e.g. speech-act) to help classify messages
  - Sentiment analysis
- Topic modelling
- Different combination of above techniques and ML techniques to help classify:
  - user comments into **bug reports, feature requests**, polarity of **sentiments**
  - classify various user groups, with different characteristics and needs
- Knowledge-based techniques
  - Ontology-based similarity for clustering feedbacks / identifying feedback related to specific requirements
Analysis techniques - Explicit user feedback

Speech-act based analysis technique*

• Speech-acts\(^1\) (rs: when we speak we affect the behaviour of the audience)

• NLP tools support the analysis of text to discover speech-acts
  • Part-Of-Speech taggers, key words
  • lexico-syntactic rules for each speech-act

---

\(^1\)Austin (1962), Searle (1969), Bach and Harnish (1979)

*Morales-Ramirez, Perini, Ceccato CAISE-forum14
Analysis techniques - Explicit user feedback
Speech-act based analysis technique

Morales-Ramirez, Kifetew, Perini, CAiSE17 and IS journal 2018
Examples of Speech-act based analysis results

RQ: Can the speech-acts be used as parameters to classify defect reports, and feature or enhancement requests?

**AOO: Using the 43 parameters.**

<table>
<thead>
<tr>
<th></th>
<th>RF</th>
<th>J48</th>
<th>SMO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F-M</td>
</tr>
<tr>
<td></td>
<td>0.79</td>
<td>0.81</td>
<td>0.84</td>
</tr>
<tr>
<td></td>
<td>0.76</td>
<td>0.80</td>
<td>0.81</td>
</tr>
</tbody>
</table>

**AOO: using 25 parameters (no speech acts).**

<table>
<thead>
<tr>
<th></th>
<th>RF</th>
<th>J48</th>
<th>SMO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P</td>
<td>R</td>
<td>F-M</td>
</tr>
<tr>
<td></td>
<td>0.79</td>
<td>0.81</td>
<td>0.81</td>
</tr>
<tr>
<td></td>
<td>0.74</td>
<td>0.78</td>
<td>0.82</td>
</tr>
</tbody>
</table>

- **Apache Open Office (AOO) dataset**
  - user feedback gathered from the AOO issue tracking system
  - 161K textual comments (2001-2017)

- **Parameters**
  - E.g. number of informative / responsive / requestive and assertive expressions, attach / logFile / urlLink

- **3 ML algorithms in WEKA**
  - Random Forest
  - J48
  - SMO

---

1[https://www.cs.waikato.ac.nz/ml/weka/](https://www.cs.waikato.ac.nz/ml/weka/)

*Morales-Ramirez, Kifetew, Perini, IS journal 2018. IS journal 2018*
Explicit and Implicit User feedback – Combined Analysis

Case Study: iESA (home energy savings apps), hundreds of users;
4-month period: both implicit and explicit feedback collected

1st workshop phase: 31 feedback messages only -> 16 relevant feedback -> 10 requirements
   E.g R1: The user should be able to see also in the Android app the meter readings of all the meters, independently of the status of the meter

2nd workshop phase: 16 relevant feedback messages + 2,164 clicks (out of 957K clicks) -> 1 new requirement and 4 requirements (among the 10 previously identified) were changed

*see for example, Oriol et al. RE18
Data-driven RE Ingredients

3. Decision-making needs
Data-driven RE decision-making

- Is this user feedback expressing a bug report or a request for a new requirement (feature)?
- Which functionalities (features) are users talking about most?
- Which functionalities (features) are perceived positively and which are perceived negatively?
- Which requirements are more important?
- Which functionalities are mostly used?
- How many users are affected by a feature request or a pointed out problem?
- Which requirements should be implemented in the next software release?
- What do different type of users needs?
  - e.g. classify various user groups; with different characteristics and needs
- What do users say about a given feature?
- ...

2 Note: See also *Software Analytics* as defined by Buse & Zimmermann, ICSE 2015
Data-driven RE decision-making

- Is this user feedback expressing a bug report or a request for a new requirement (feature)?
- Which functionalities (features) are users talking about most?
- Which functionalities (features) are perceived positively and which are perceived negatively?
- Which requirements are more important?
  - E.g. given a set of requirements prioritize them taking into account user value as expressed in feedback
- Which functionalities are mostly used?
- How many users are affected by a feature request or a pointed out problem?
- Which requirements should be implemented in the next software release?
- What do different type of users needs?
  - E.g. classify various user groups; with different characteristics and needs
- What do users say about a given feature?
- ...

- What issues (bug fixing / feature enhancement) shall we implement in the next release?

\(^2\) Note: See also *Software Analytics* as defined by Buse & Zimmermann, ICSE 2015
The SUPERSEEDE loop:
Enabling a data-driven engineering process

• **collect** end-users’ feedback and runtime data in an efficient, scalable and adaptable way
• perform an integrated **analysis** of the collected data
• **support decision-making** in the **evolution** and **runtime adaptation of services** and applications based on user’s feedback and contextual data
• **enact** the decisions made
SUPERSEDE case study
Data-driven RE for software evolution

Pre-configuration
- Existing User feedback, and requirements
- Domain ontology & ECA rules
- Customise SUPERSEDE for my application domain
- Development team info and JIRA configured
  Criteria, roles, dependencies
- Pending issues in planner,
  Resources, skills

Run-time
SUPERSEDE case study
Data-driven RE for software evolution

Pre-configuration
- Existing User feedback, and requirements
- Domain ontology & ECA rules

Alert generated from feedback - similar issues
Issues reflecting users’ perspective, business, Technical

Customise SUPERSEDE for my application domain
Which issues should I implement in the next release?

Development team info and JIRA configured
Criteria, roles, dependencies

Pending issues in planner, Resources, skills

Run-time
- New rules

New User feedback

FILTER

SUPERSEDE webinar: https://www.dropbox.com/s/cbiw5yf2z1q2vm8/se4sa-WebinarSUPERSEDE060318_v2_1.mov?dl=0
Feedback Gathering & Monitoring
FAME - Big data mngt. & data analytics architecture

*Oriol et al., RE18
What’s behind - DMGame and REPlan

- **DMGame**
  - Automated reasoning: Analytic Hierarchy Process (AHP)
  - Genetic Algorithms (GA)
  - Gamification to foster decision-makers engagement
    - Gamification mechanisms, e.g. Progress bar, Pointsification (gaining points if quick and if accurate)

- **RePlan Optimizer**
  - Optimization algorithms from the JMetal library
The SUPERSEDE toolsuite

- Integration in JIRA (© Atlassian)
- Software components - available for download at [https://github.com/supersed-e-project](https://github.com/supersed-e-project)
SUPERSEDE Methodology - web-based tool

- A methodology for using the SUPERSEDE approach based situational method engineering principles*


*Franch, et al. CAISE 2018
Outline

✓ Setting the scene
✓ Data-driven Requirements Engineering
  - The SUPERSEDE way
• Conclusion:
  - Open research challenges
  - AI & RE
• Special thanks
Open Research Challenges
Research challenges – Data-driven RE

• What data and how much?
• What requirements / software engineering decisions?
  - Adopt a decision-oriented, rather than an data analytics oriented perspective
  - Integrate analytics methods in sw development platforms & associated RE/SE methods

• User feedback collection:
  - Motivating users to provide feedback, taking into account their diversity (e.g. Loyal and passionate ...Impact seekers)
    • *digital motivation*: adaptive to the context and profile of user

• Quality of user feedback:
  - Subjectiveness (no rationale behind a feedback), conflicting contributions
    • Mixing Pull and Push feedback mechanisms

• Methodological:
  - Misuse (i.e. extracting personal information of users)
    • *Privacy-aware; transparency*
  - Extending / improving analysis of /and management of user input within an open science approach
AI & RE in Continuous Delivery
AI & RE in Continuous Delivery

Continuous delivery ... challenging RE

- Requirements artefacts (feature, test cases)
- Traceability
- ....
- Continuous assessment of the effect of a new software release
- Collecting and analysing usage data and user feedback at support of informed decisions
- Effective (multi-criteria) requirement prioritization

- Many RE artefacts use textual natural language
  - NLP techniques +
    - Extracting requirements relevant info -> “informed decision-making”
    - Automatic extraction and maintenance of traceability links -> ”tool-supported change management”

- RE is Multi-objective decision making
  - Search-based algorithms
  - Knowledge representation and reasoning at support of “decision transparency”

- RE uses models
  - Exploiting data-driven techniques to validate, evolve

AIRE - Artificial Intelligence for Requirements Engineering Workshop - 5-th ed. At RE’18
RE & AI?

RE for AI?
RE’19 Special Theme:
RE and Collective Intelligence in the Days of AI

- A unique opportunity for RE research to lead a **paradigm shift** that places human and society at the forefront of the design of AI systems
  - helping address **ethical considerations** in the design, use, and misuse of intelligent systems;
  - “how do we collaborate among various groups of people with different perspectives and expectations so that we end up designing the **right system** with the right set of behaviors?”
- **as opposed to** “how do we make a system intelligent with the latest technical achievements?”
Outline

✓ Setting the scene

✓ Data-driven Requirements Engineering
  - The SUPERSEDE way

✓ Conclusion:
  - Open research challenges
  - AI & RE

• Special thanks
Special thanks: ... 9 rich months

... don’t give up! :-}
Thanks to the SUPERSEDE Team
Thank you for your attention

Questions?
Main References

- Raymond P. L. Buse, Thomas Zimmermann: Information needs for software development analytics. ICSE 2012: 987-996
- The SUPERSEDE project, www.supersede.eu (Scientific Publications)