Survey of approaches for System Requirements

Présentation Journées du GDR GPL – GT IE

18/06/2021

S. Ebersold

The Role of Formalism in System Requirements

- ACM Computing Surveys, Volume 54, Issue 5, June 2021, pp 1–36
  https://dl.acm.org/doi/10.1145/3448975
- ArXiv : https://arxiv.org/abs/1911.02564

The Role of Formalism in System Requirements

- Exigences vs. spécifications
- Exemple fil rouge
- Approches étudiées
- Critères d'évaluation
- Évaluation - résultats
- Déductions
- Conclusion

Choix des 23 approches étudiées

REQTIFY

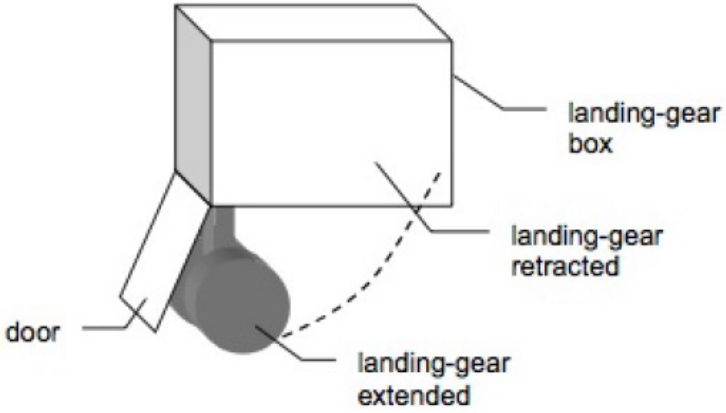
Eiffel Software

- Largement utilisées (disponibles dans le commerce ex. : Doors)
- Largement connues
- Influentes
- Possédant d'autres caractéristiques intéressantes

Name	Category	References	Section
Requirements Grammar*	Natural language	[118]	4.1.1
Relax		[139]	4.1.2
Stimulus		[58]	4.1.3
NL to OCL*		[49]	4.1.A
NL to STD*		[4]	4.1.B
NL to OWL*		[62]	4.1.4
Doors	Semi-formal	[50]	4.2.1
Reqtify		[124]	4.2.1
KAOS		[135]	4.2.2
URN		[7]	4.2.A
SysML		[94]	4.2.3
URML		[12]	4.2.B
Petri Nets	Automata- or graph-based	[105]	4.3.A
Statecharts		[41]	4.3.1
Problem Frames		[57]	4.3.2
FSP/LTSA		[67]	4.3.3
FORM-L		[91]	4.3.4
Event-B	Mathematical notation	[2]	4.4.1
VDM		[13]	4.4.A
Process Algebra		[47, 77]	4.4.B
Alloy		[56]	4.4.2
Tabular Relations		[103]	4.4.C
Multirequirements	Seamless	[74, 88]	4.5

Etude de cas : Landing Gear System (Boniol et al. 2014)

- (R11bis) If the landing gear command handle has been pushed down and stays down, then eventually the gears will be locked down and the doors will be seen closed.
- (R12bis) If the landing gear command handle has been pushed up and stays up, then eventually the gears will be locked retracted and the doors will be seen closed.
- (R21) If the landing gear command handle remains in the down position, then retraction sequence is not observed.
- (R22)If the landing gear command handle remains in the up position, then outgoing sequence is not observed.



Approches en langage naturel

Expriment les exigences en anglais ou dans une autre langue humaine, mais peuvent restreindre le degré de "naturalité" du texte.

Exemple : Relax

R11bis: The gear SHALL be locked down and the doors SHALL be closed AS EARLY AS POSSIBLE AFTER the landing gear command handle has been pushed down and stays down.

R12bis: The gear SHALL be locked retracted and the doors SHALL be closed AS EARLY AS POSSIBLE AFTER the landing gear command handle has been pushed up and stays up.

R21: The retraction sequence SHALL not be observed AS EARLY AS POSSIBLE AFTER the command handle remains in down position.

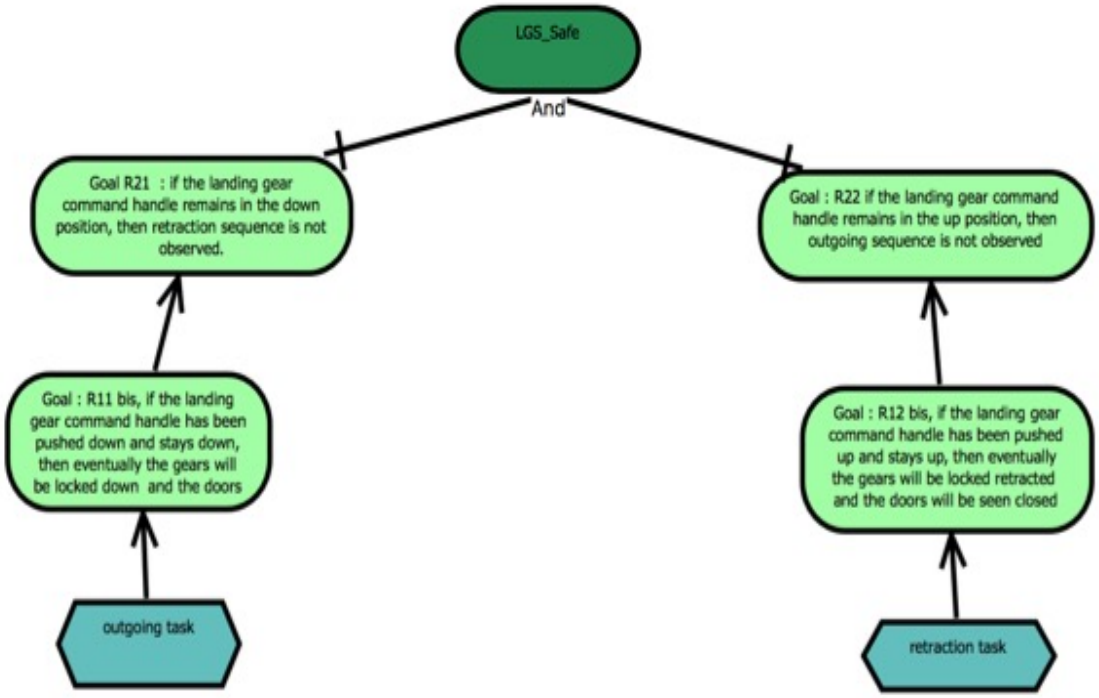
R22: The outgoing sequence SHALL not be observed AS EARLY AS POSSIBLE AFTER the command handle remains in up position.

Approches semi-formelles

Définissent un langage d'exigences précis (codifient la forme des exigences), qui n'est ni une notation mathématique, ni dérivé d'un langage de programmation

Exemple :

User Requirements Notation (URN)



Approches à automates ou à graphes

Reposent sur des notations basées sur les automates ou la théorie des graphes

Fournissent généralement un support graphique

Peuvent laisser la base mathématique implicite

Exemple : FormL

```
requirement r11 is
    after (lgsHandle becomes down)
        and (not failure)
        and not (allGearsDown and allDoorsClosed)
    within 15*s
    check (allGearsDown and allDoorsClosed) becomes true
        or lgsHandle becomes up
        or failure becomes true;
```

Approches mathématiques

Reposent sur des formalismes mathématiques (base théorique généralement logique mathématique)

Exemple : VDM-SL

```
operations
  extension_sequence()
    ext wr gears
        wr doors
        rd handle
    pre handle =<down>
    post handle =<down>  => (gears =<extended>  and doors =<closed>);

  retraction_sequence()
    ext wr gears
        wr doors
        rd handle
    pre handle =<up>
    post handle =<up>  => (gears =<retracted>  and doors =<closed>);
end LGS
```

Approches sans couture et de programmation

Intègrent étroitement les exigences aux autres tâches logicielles (conception, implémentation...), en utilisant un langage de programmation comme notation

Exemple : Seamless object-oriented requirements (SOOR)

```
class
  R22
inherit
  ABSENCE_BEFORE [LGS, GEAR_EXTENDING, HANDLE_DOWN]
  LGS_REQUIREMENT
end
```

```
frozen verify (system: LGS)
  -- (from ABSENCE_BEFORE)
 do
  from
   timer := time_boundary
  invariant
   p_does_not_hold_or_else_r_holds: not ({GEAR_EXTENDING}).default.holds (system) or
       ↪ else ({HANDLE_DOWN}).default.holds (system)
  variant
   timer
  until
   ({HANDLE_DOWN}).default.holds (system)
  loop
   iterate (system)
  end
end
```

Critères d’évaluation

- Audience : niveau d'expertise attendu des personnes qui utiliseront les exigences.
  - F (formation en méthodes formelles), M (connaissances générales en mathématiques), S (formation spécifique requise), N (aucune formation particulière attendue)
- Niveau d'abstraction : niveau de détail (des propriétés du système décrit) que les exigences peuvent ou doivent couvrir
  - (H (élevé), L (Faible), B (les deux))
- Méthode associée : évalue si l'approche comprend une méthodologie complète pour guider le processus de définition des exigences
- Outil support : couvre la disponibilité d'outils pour supporter l'approche, par opposition aux approches qui sont uniquement conceptuelles.
- Support de la traçabilité : évalue si l'approche traite le suivi des relations entre les éléments des exigences et leurs contreparties dans la conception, le code et les autres artefacts du projet.

Critères d’évaluation

- Couverture : vise à déterminer si l'approche couvre uniquement la description des propriétés fonctionnelles du système ou si elle s'étend aux propriétés non fonctionnelles (affectant des aspects comme les performances et la sécurité).

- Portée : description de l'environnement et des contraintes qu'il impose Vs. Description du système uniquement (Distinction Jakson-Zave).
  - S (Système), B (les deux)

- Vérifiabilité : évalue si l’approche supporte la possibilité de vérifier formellement les propriétés des exigences résultantes.

- Définition sémantique : évalue la disponibilité et la portée d'une définition précise (si possible, formelle) de l'approche.

Evaluation

A summary of the results

	System vs Environment	Prerequisites	Level of abstraction	Associated method	Traceability support	Non-functional req. support	Semantic definition	Tool support	Verifiability
NL to OCL	S	S	H	✗	✗	✗	✓	✓	✓
NL to OWL	B	S	H	✓	✓	✓	✓	✗	✗
NL to STD	B	S	H	✓	✓	✓	✗	✗	✓
Relax	B	N	H	✗	✓	✓	✓	✓	✓
Stimulus	B	N	H	✗	✗	✗	✓	✓	✓
Requirements Grammar	B	N	H	✗	✗	✓	✗	✓	✗
Doors	B	N	L	✗	✓	✓	✗	✓	✗
Reqtify	B	N	L	✗	✓	✓	✗	✓	✗
KAOS	B	S	B	✓	✓	✓	✓	✓	✗
URN	S	S	H	✓	✓	✓	✗	✓	✗
SysML	S	S	H	✗	✓	✓	✗	✓	✗
URML	B	S	H	✗	✓	✓	✓	✓	✗
FORM-L	B	S	H	✗	✗	✗	✓	✓	✓
FSP/LTSA	S	S	H	✓	✗	✗	✓	✓	✓
Petri Nets	S	S	H	✗	✗	✗	✓	✓	✓
Problem Frames	B	S	H	✓	✗	✗	✓	✓	✓
Statecharts	S	S	H	✓	✗	✗	✓	✓	✓
Alloy	S	F	H	✗	✗	✗	✓	✓	✓
Event-B	B	F	B	✓	✗	✗	✓	✓	✓
Tabular Relations	B	M	H	✓	✗	✗	✓	✗	✗
VDM	B	F	H	✓	✗	✗	✓	✓	✓
Multirequirements	B	S	B	✓	✓	✗	✓	✓	✗
SOOR	B	N	B	✓	✓	✗	✓	✓	✓

Déductions

- Informel vs. semi-formel vs. formel ?
- Sans couture vs. conventionnel ?
- Textuel vs. graphique ?
- Quels outils supports ?
- Quelle éducation ?

Quel rôle pour les approches formelles des exigences ?

Quel rôle pour les approches formelles des exigences ?

- Plusieurs degrés de formalité appropriés dans l'énoncé des exigences pour les systèmes logiciels

- Les outils doivent constituer de véritables supports

- L’éducation des futurs ingénieurs logiciels est fondamentale

Conclusion

- Les méthodes formelles sont parfois considérées comme trop théoriques
- **Mais** L'ingénierie des exigences est essentielle à la pratique de la construction de logiciels
- **Alors** Les méthodes formelles complètent les autres techniques d'exigences, plutôt que de tenter de les remplacer

Les méthodes formelles peuvent et doivent être une aide puissante à la disposition de tout ingénieur des exigences