Logics for Terms of Services and their Usefulness for Automation

Cristian Prisacariu

Precise Modeling and Analysis group (PMA),
University of Oslo

at

FSCONS

10^{th} November 2013, Gothenburg, Sweden.

LoCo – Logics for Contracts

The project

LoCo – Logics for Contracts

The project

Target:

Terms of Services (ToS)

LoCo – Logics for Contracts

The project

Target:

Terms of Services (ToS)

Why?

Empowering people

LoCo – Logics for Contracts

The project

Target:

Terms of Services (ToS)

Why?

Empowering people

How?

Let the browser check the ToS

LoToS – Logics for Terms of Service

The project

LoToS – Logics for Terms of Service

The project

Automation for Terms of Services

- Automate the reading.
- Automated analysis (personalized).
- Automated translation.
- Automated negotiation.
- Drafting.
- Visualization.
- Summaries.
- Comparisons.

LoToS – Logics for Terms of Service

The project

Include:

- Natural Language Processing (NPL).
- Knowledge representation (KR).
- Understanding norms and actions.
- Inference engine.
- Verification of desired properties/requirements.
- Monitor.
- Open system administered by a community.
- User friendly and ease of use.

LoToS – The Intended Outcome

User's point of view

LoToS – The Intended Outcome

User's point of view

1. A new ToS text is read into a logical model for LoToS.

LoToS – The Intended Outcome

User's point of view

1. A new ToS text is read into a logical model for LoToS.
2. User's expectations/requirements/properties/rules are checked by the LoToS model checker against the new ToS.

LoToS – The Intended Outcome

User's point of view

1. A new ToS text is read into a logical model for LoToS.
2. User's expectations/requirements/properties/rules are checked by the LoToS model checker against the new ToS.
3. If the check goes through, then the ToS respects the user's requirements (privacy, economical, etc.)

LoToS – The Intended Outcome

User's point of view

1. A new ToS text is read into a logical model for LoToS.
2. User's expectations/requirements/properties/rules are checked by the LoToS model checker against the new ToS.
3. If the check goes through, then the ToS respects the user's requirements (privacy, economical, etc.)

No user input was needed up to now.

LoToS – The Intended Outcome

User's point of view

1. A new ToS text is read into a logical model for LoToS.
2. User's expectations/requirements/properties/rules are checked by the LoToS model checker against the new ToS.
3. If the check goes through, then the ToS respects the user's requirements (privacy, economical, etc.)

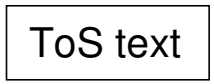
No user input was needed up to now.

4. If some requirement fails, the user is provided with an explanation.
5. Visualization and summary of the explanation/trace is needed.

The user takes the ultimate decision to accept the ToS.

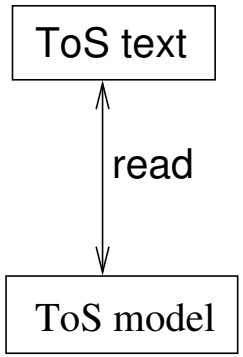
LoToS – The Intended Outcome

User's point of view



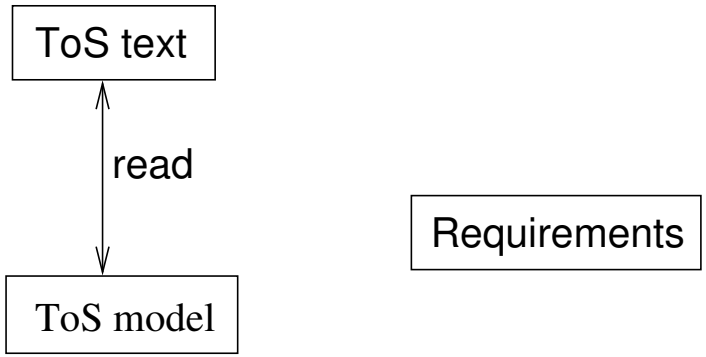
LoToS – The Intended Outcome

User's point of view



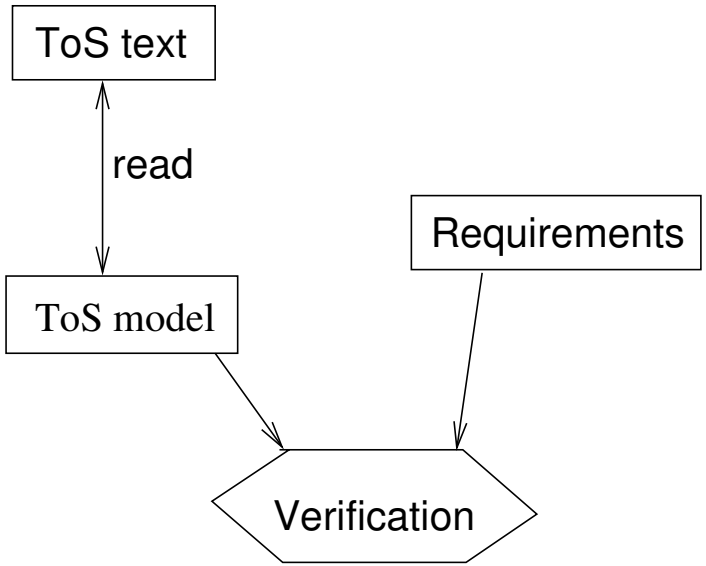
LoToS – The Intended Outcome

User's point of view



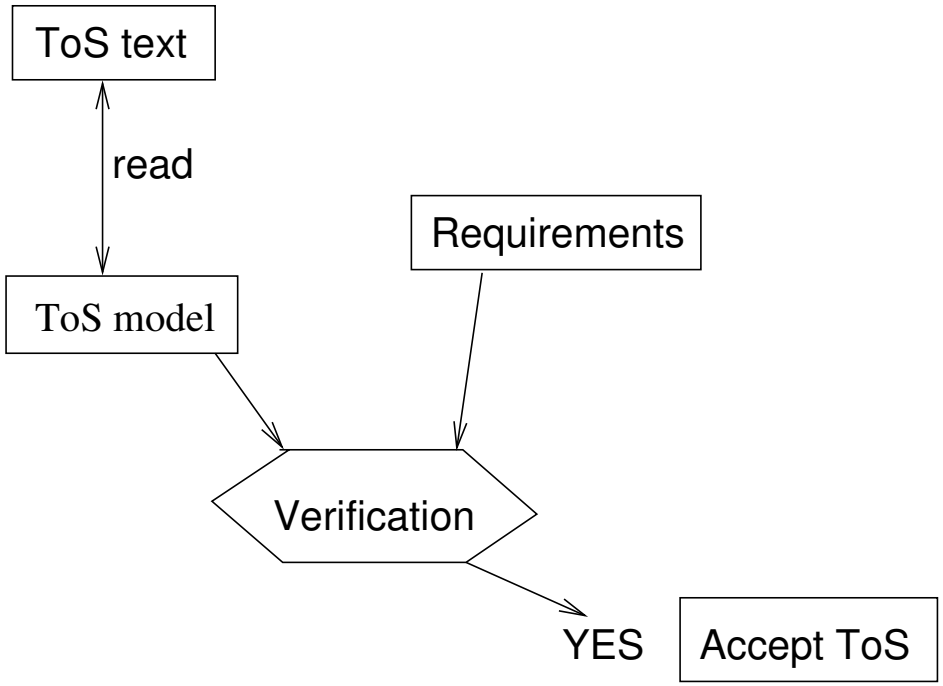
LoToS – The Intended Outcome

User's point of view



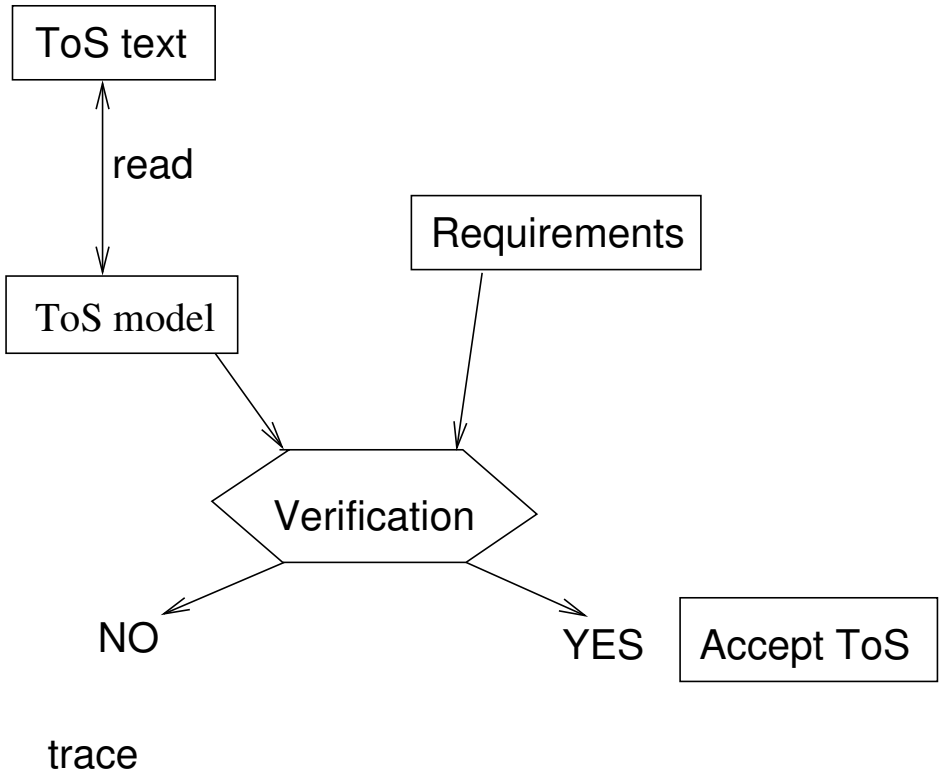
LoToS – The Intended Outcome

User's point of view



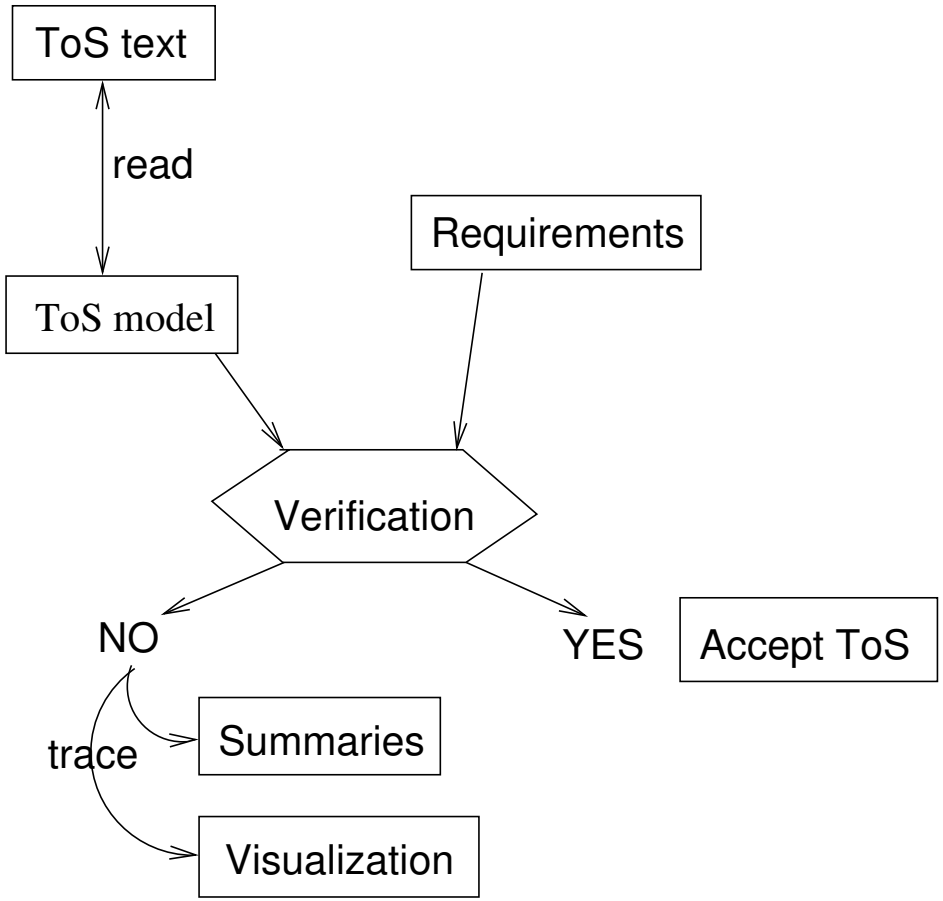
LoToS – The Intended Outcome

User's point of view



LoToS – The Intended Outcome

User's point of view



User's point of view

What can go wrong

User's point of view

What can go wrong

LoToS is inevitably human-assisted.

User's point of view

What can go wrong

LoToS is inevitably human-assisted.

1. When reading the ToS text
   a passage cannot be parsed, or is ambiguous for CNL parser

User's point of view

What can go wrong

LoToS is inevitably human-assisted.

1. When reading the ToS text
   a passage cannot be parsed, or is ambiguous for CNL parser

- User is notified for
  – help with the parsing rules
  – disambiguation
  – or to ignore that part of the ToS

User's point of view

What can go wrong

LoToS is inevitably human-assisted.

1. When reading the ToS text
   a passage cannot be parsed, or is ambiguous for CNL parser

   - User is notified for
     – help with the parsing rules
     – disambiguation
     – or to ignore that part of the ToS
   - A non-expert user may access the on-line LoToS system
     – where the present ToS hopefully/probably exists
     – expert users from the community took care to disambiguate
     – any choices for a ToS can be visualized by the user

User's point of view

What can go wrong – 2

2. Requirements
   are difficult to define.

User's point of view

What can go wrong – 2

2. Requirements
   are difficult to define.

- templates predefined by the community experts can be taken a priori and filled in for the present ToS
- for ToS existing in the on-line LoToS
  predefined requirements can readily be taken
- The user administers a personal wallet for requirements.

  (Care needs to be taken by the community for the available requirements, so to avoid clutter.)

User's point of view

What can go wrong – 3

2. Requirements
   who checks them?

User's point of view

What can go wrong – 3

2. Requirements
   who checks them?

- model-checking can be computational intensive.
- for existing ToS any requirements in the on-line LoToS should have already been checked

User's point of view

What can go wrong – 3

2. Requirements
   who checks them?

- model-checking can be computational intensive.
- for existing ToS any requirements in the on-line LoToS should have already been checked
- Any new requirements are a model-checking problem
  – that can be solved on the user's machine
  – or through the community's distributed model-checking system

User's point of view

What can go wrong – 4

With quantities, deadlines, and other quantifiable notions like privacy, satisfying requirements can be fuzzy, i.e., on a scale range.

User's point of view

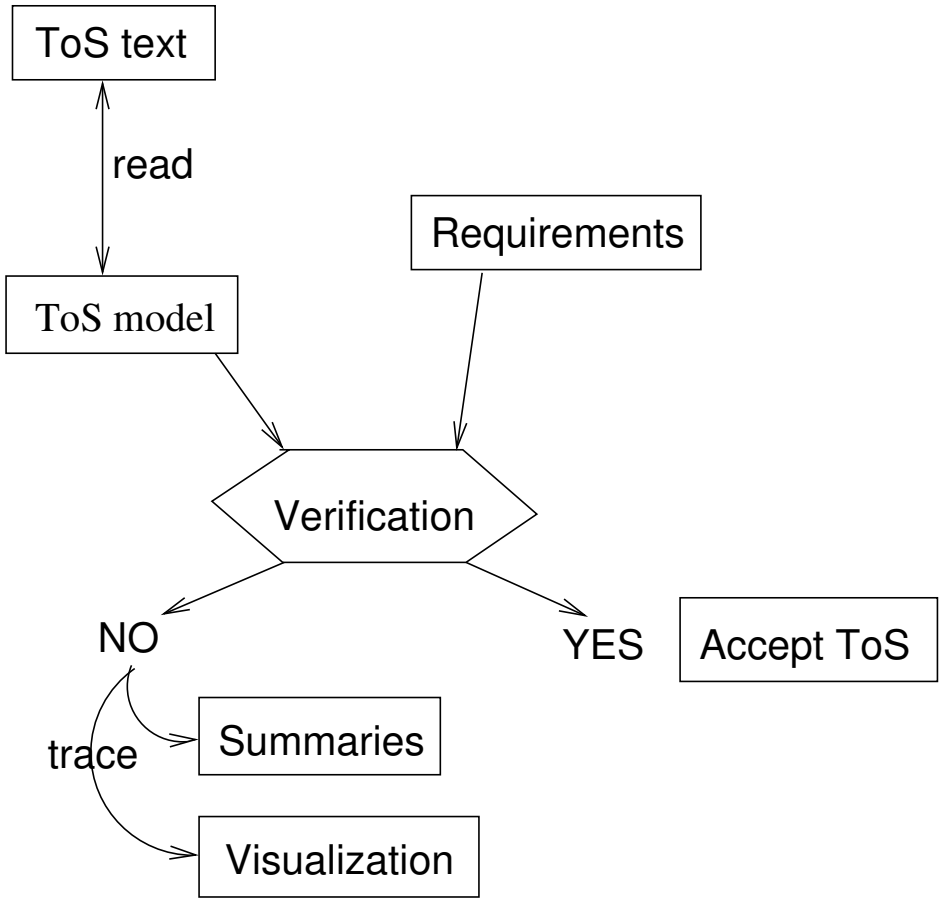
What can go wrong – 4

With quantities, deadlines, and other quantifiable notions like privacy, satisfying requirements can be fuzzy, i.e., on a scale range.

- Thresholds can be used to determine when to signal failure.
- Otherwise, the verification can return a quantitative evaluation.
  The user decides if the requirements evaluated against the ToS have a reasonable outcome to allow acceptance.
  – Explanations are more difficult.
  – Visualization could help (LoToS needs information designers).

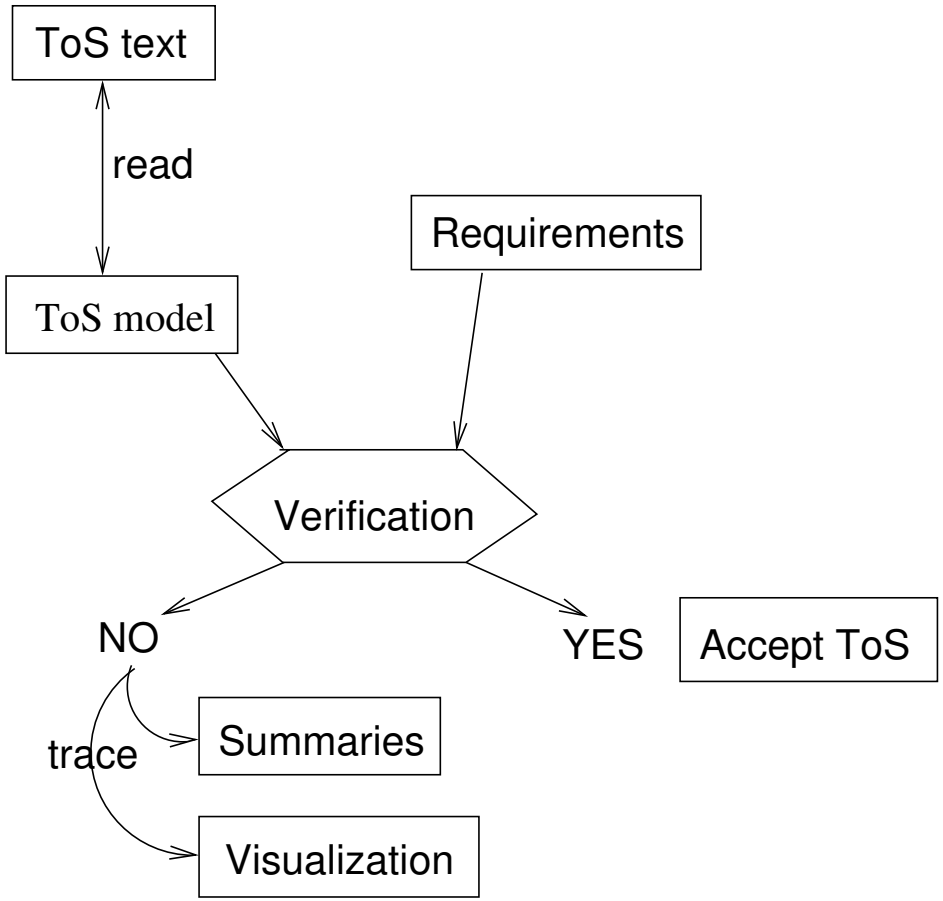
User's point of view

help from on-line LoToS



User's point of view

help from on-line LoToS

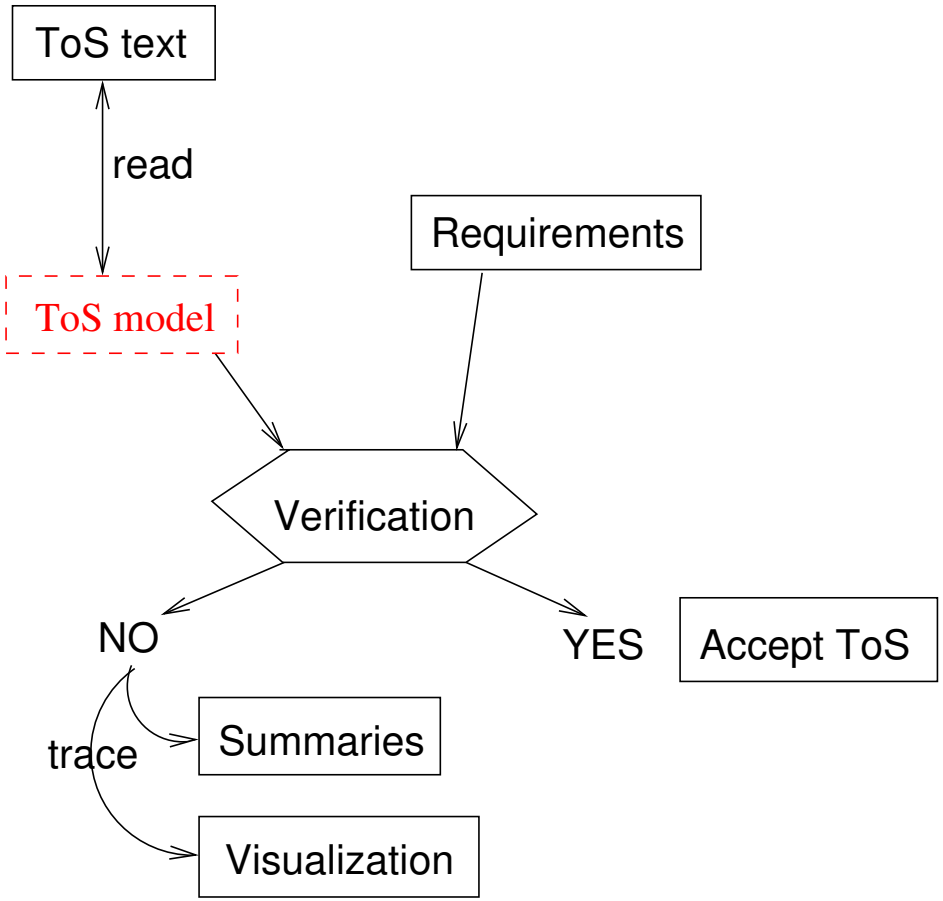


On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

User's point of view

help from on-line LoToS

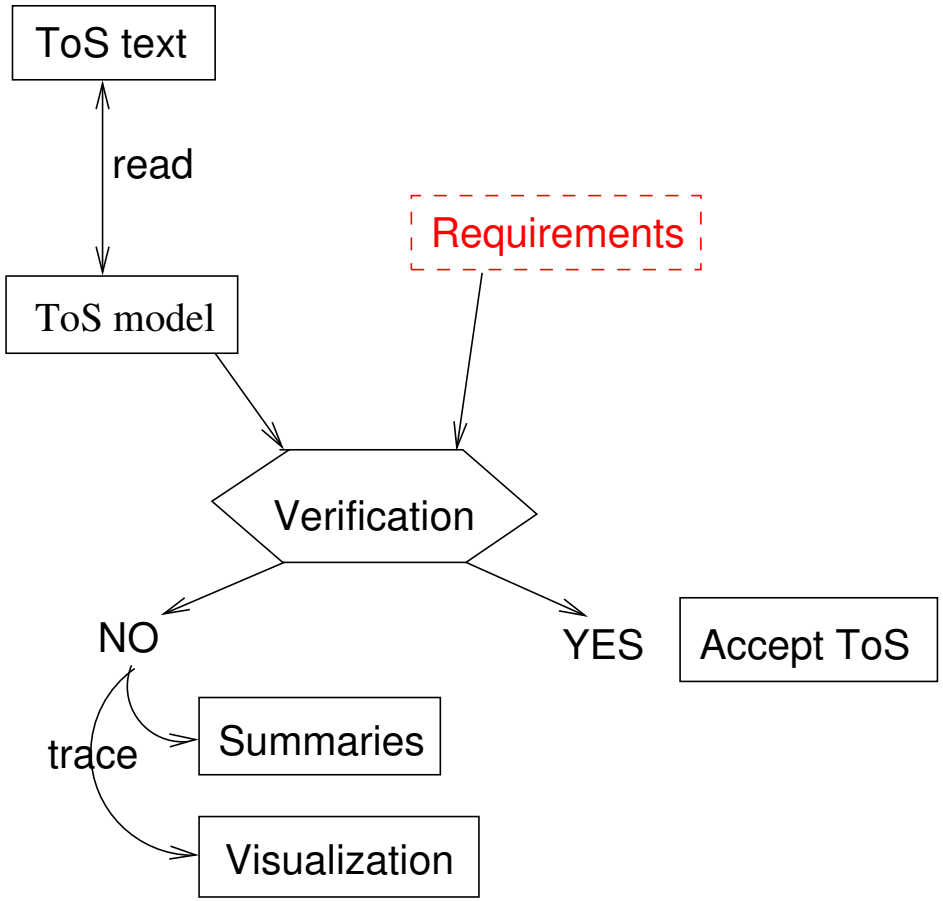


On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

User's point of view

help from on-line LoToS

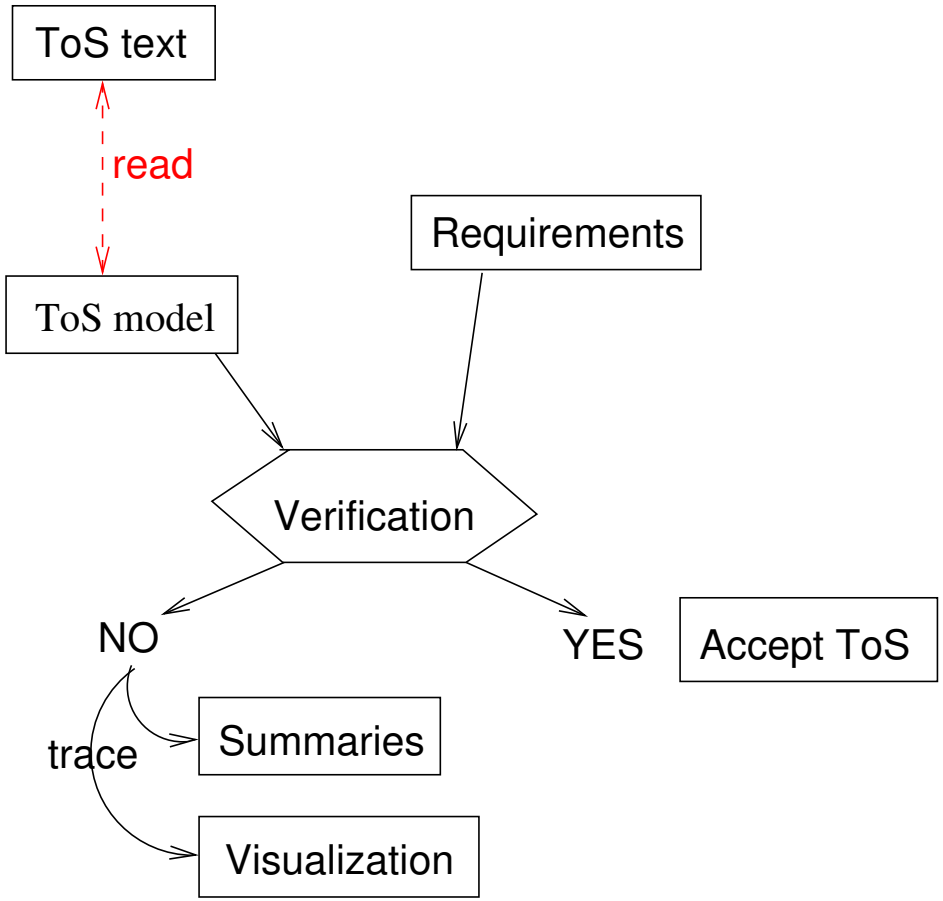


On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

User's point of view

help from on-line LoToS

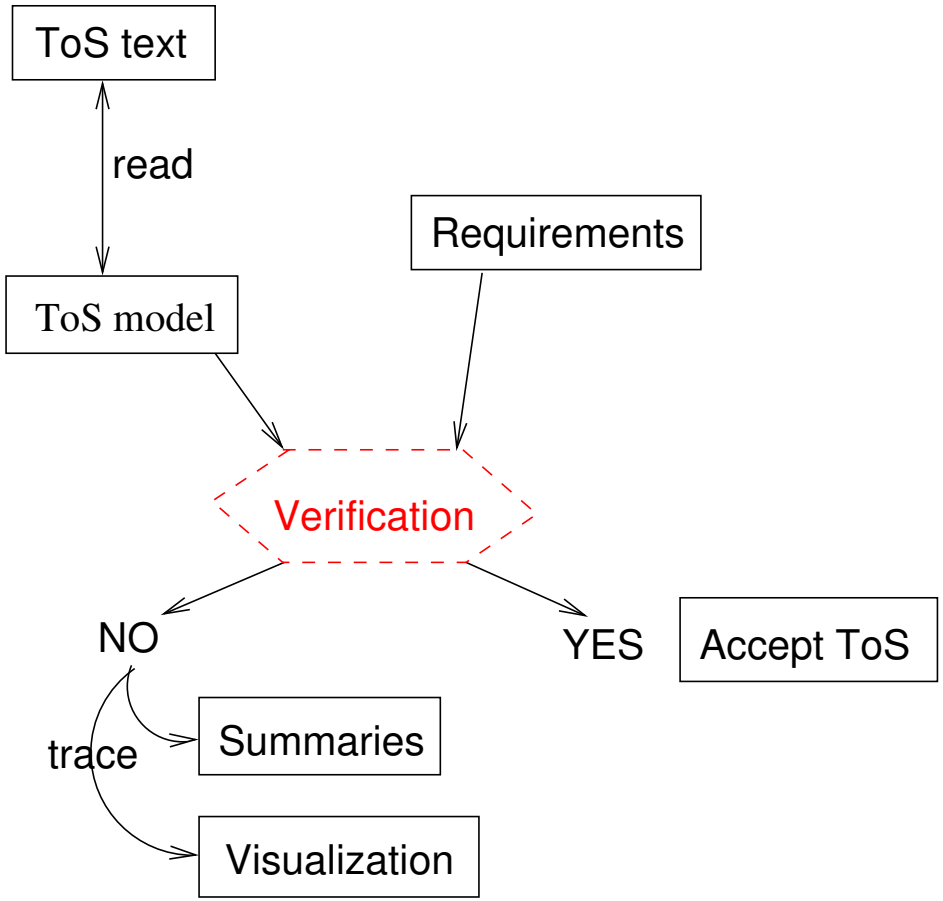


On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

User's point of view

help from on-line LoToS



On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

Technologies and Logics behind LoToS

to be detailed further

- Controlled Natural Language (CNL) – based on first-order logic (FOL)
  for reading

Technologies and Logics behind LoToS

to be detailed further

- Controlled Natural Language (CNL) – based on first-order logic (FOL)
  for reading
- Grammatical Framework (GF) – a functional language
  for translations

Technologies and Logics behind LoToS

to be detailed further

- Controlled Natural Language (CNL) – based on first-order logic (FOL)
  for reading
- Grammatical Framework (GF) – a functional language
  for translations
- Knowledge Representation (KR), legal ontology – fragments of FOL
  for capturing the right legal terminology

Technologies and Logics behind LoToS

to be detailed further

- Controlled Natural Language (CNL) – based on first-order logic (FOL)
  for reading
- Grammatical Framework (GF) – a functional language
  for translations
- Knowledge Representation (KR), legal ontology – fragments of FOL
  for capturing the right legal terminology
- Deontic logics (DL) and Logics of actions (PDL)
  for understanding norms and actions

Technologies and Logics behind LoToS

to be detailed further

- Controlled Natural Language (CNL) – based on first-order logic (FOL)
  for reading
- Grammatical Framework (GF) – a functional language
  for translations
- Knowledge Representation (KR), legal ontology – fragments of FOL
  for capturing the right legal terminology
- Deontic logics (DL) and Logics of actions (PDL)
  for understanding norms and actions
- Rule-based reasoning, model-checking, temporal logics (TL)
  for verification of requirements

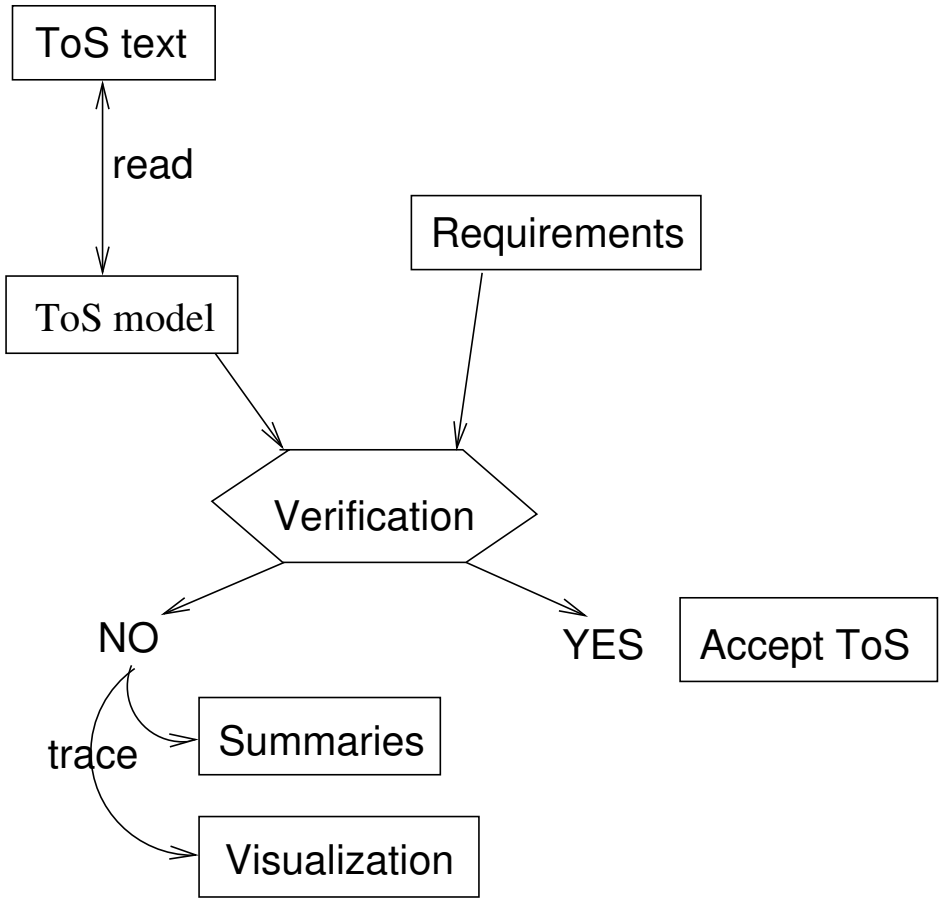
Technologies and Logics behind LoToS

to be detailed further

- Controlled Natural Language (CNL) – based on first-order logic (FOL)
  for reading
- Grammatical Framework (GF) – a functional language
  for translations
- Knowledge Representation (KR), legal ontology – fragments of FOL
  for capturing the right legal terminology
- Deontic logics (DL) and Logics of actions (PDL)
  for understanding norms and actions
- Rule-based reasoning, model-checking, temporal logics (TL)
  for verification of requirements
- Weighted logics, real-time logics, multi-valued/fuzzy logics
  for non-Boolean notions like deadlines, quantities

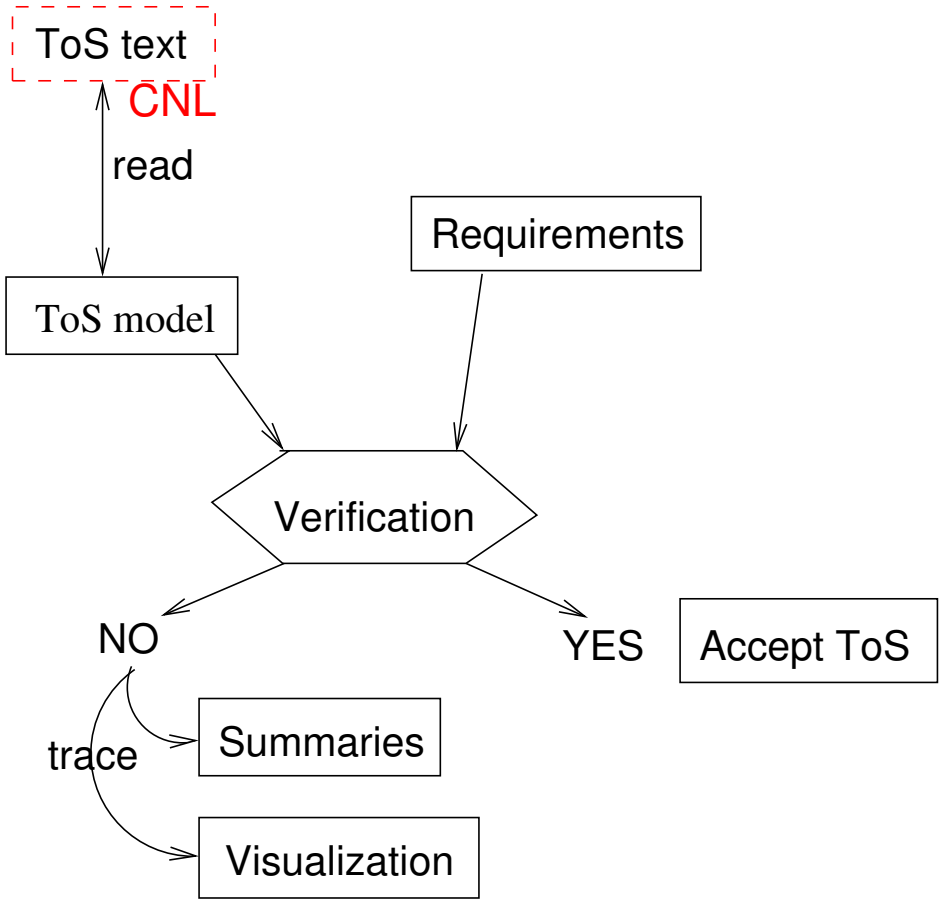
Technologies and Logics

where do they fit in LoToS ?



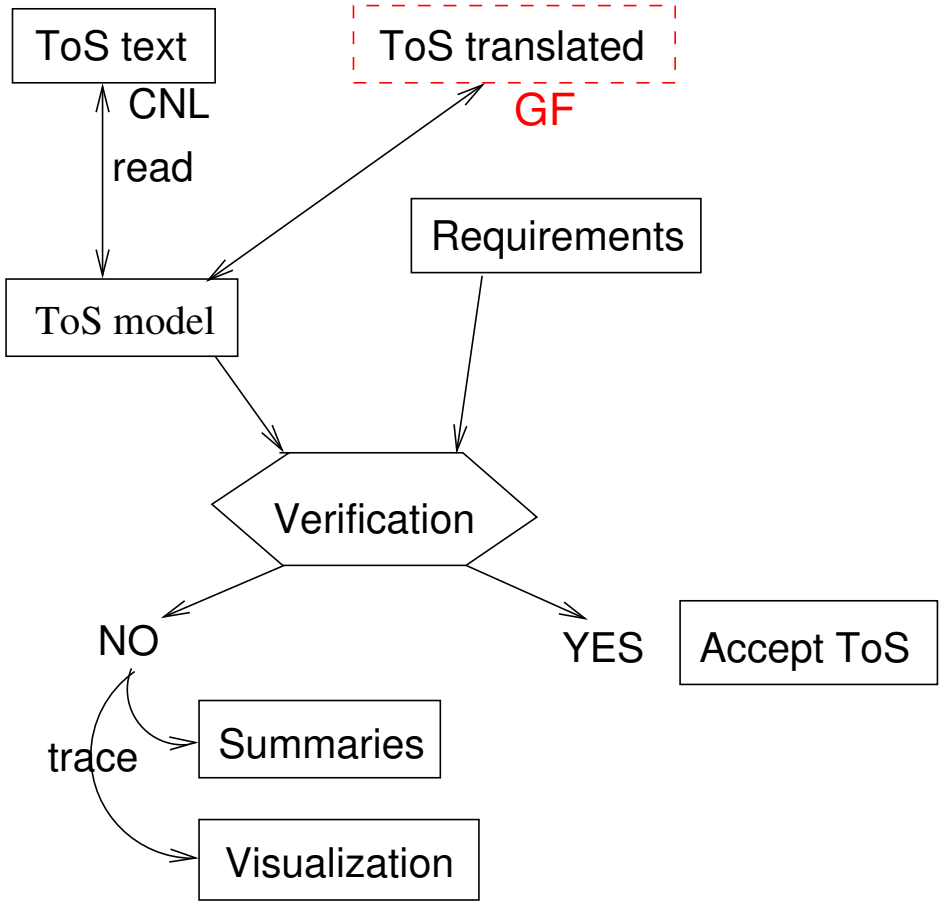
Technologies and Logics

where do they fit in LoToS ?



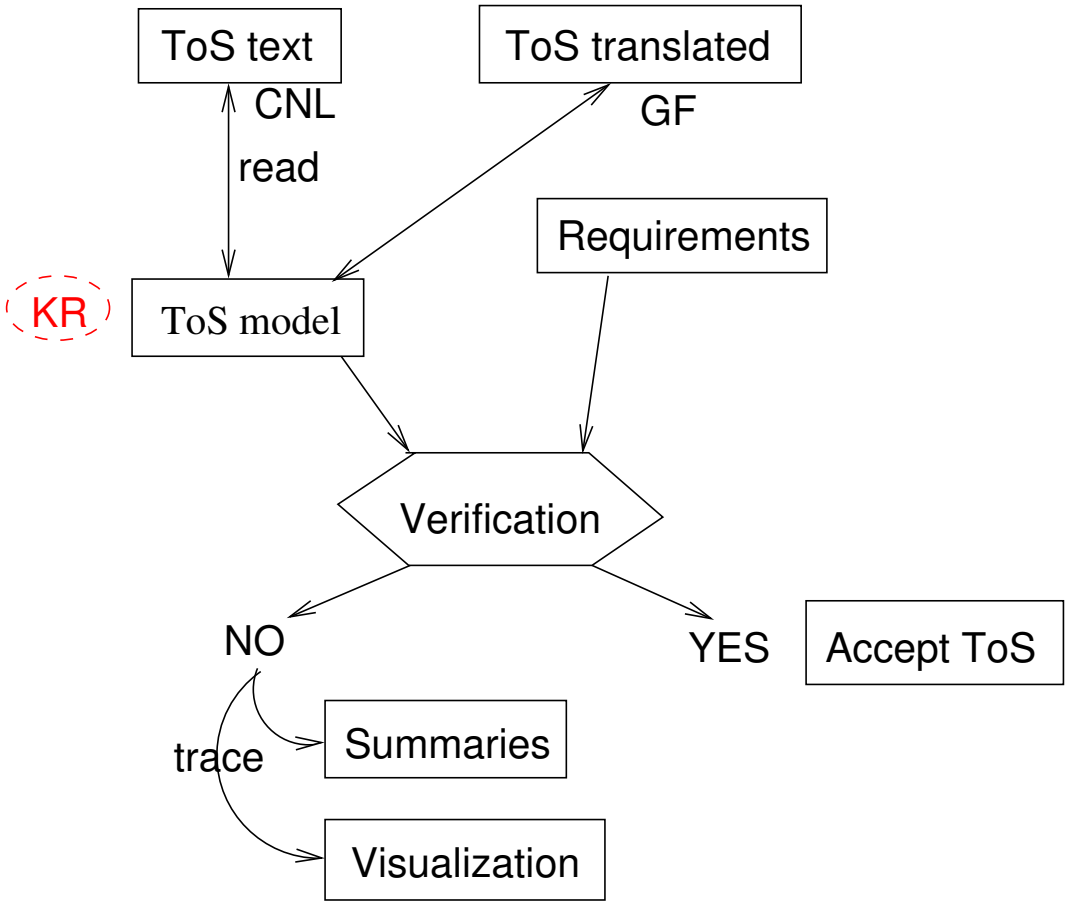
Technologies and Logics

where do they fit in LoToS ?



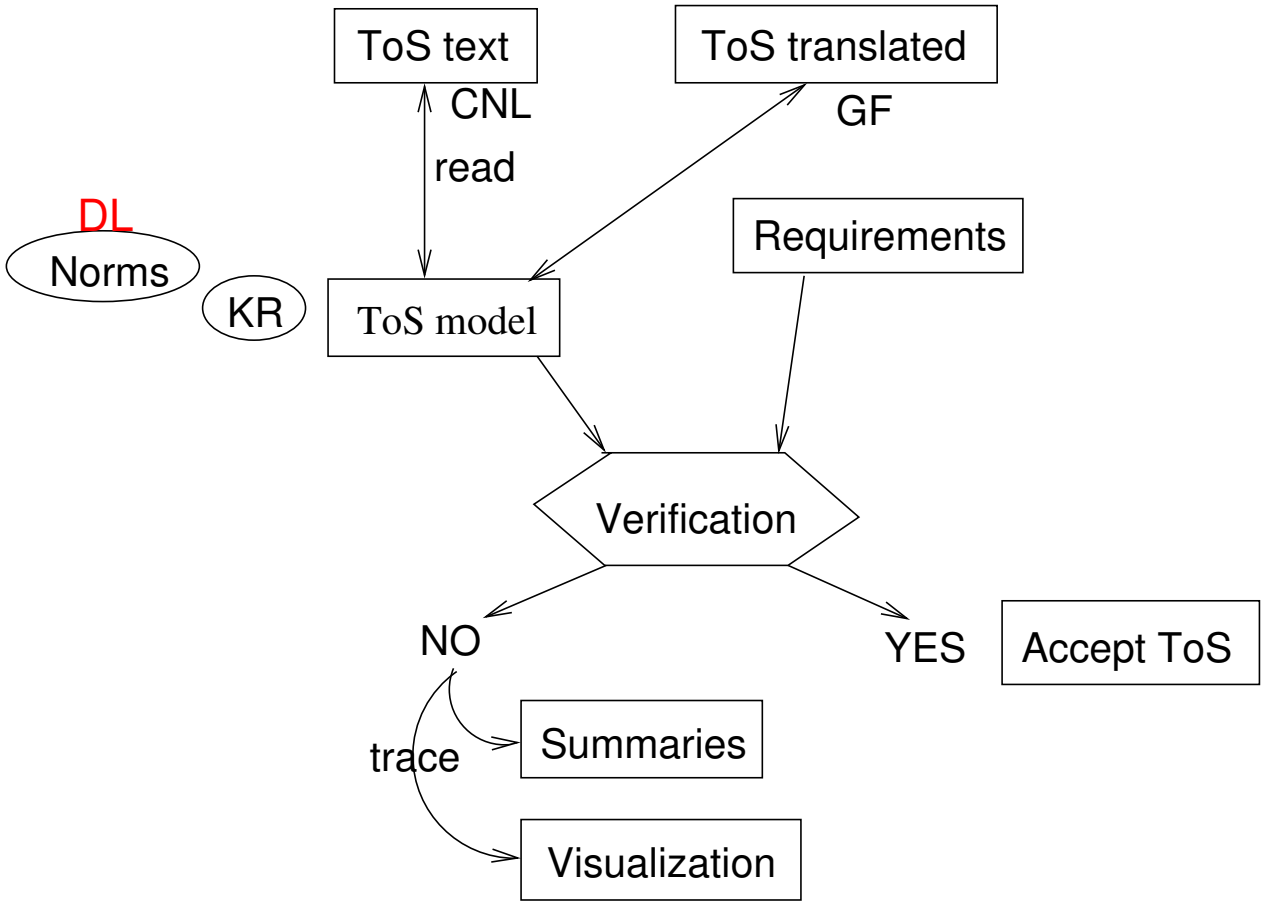
Technologies and Logics

where do they fit in LoToS ?



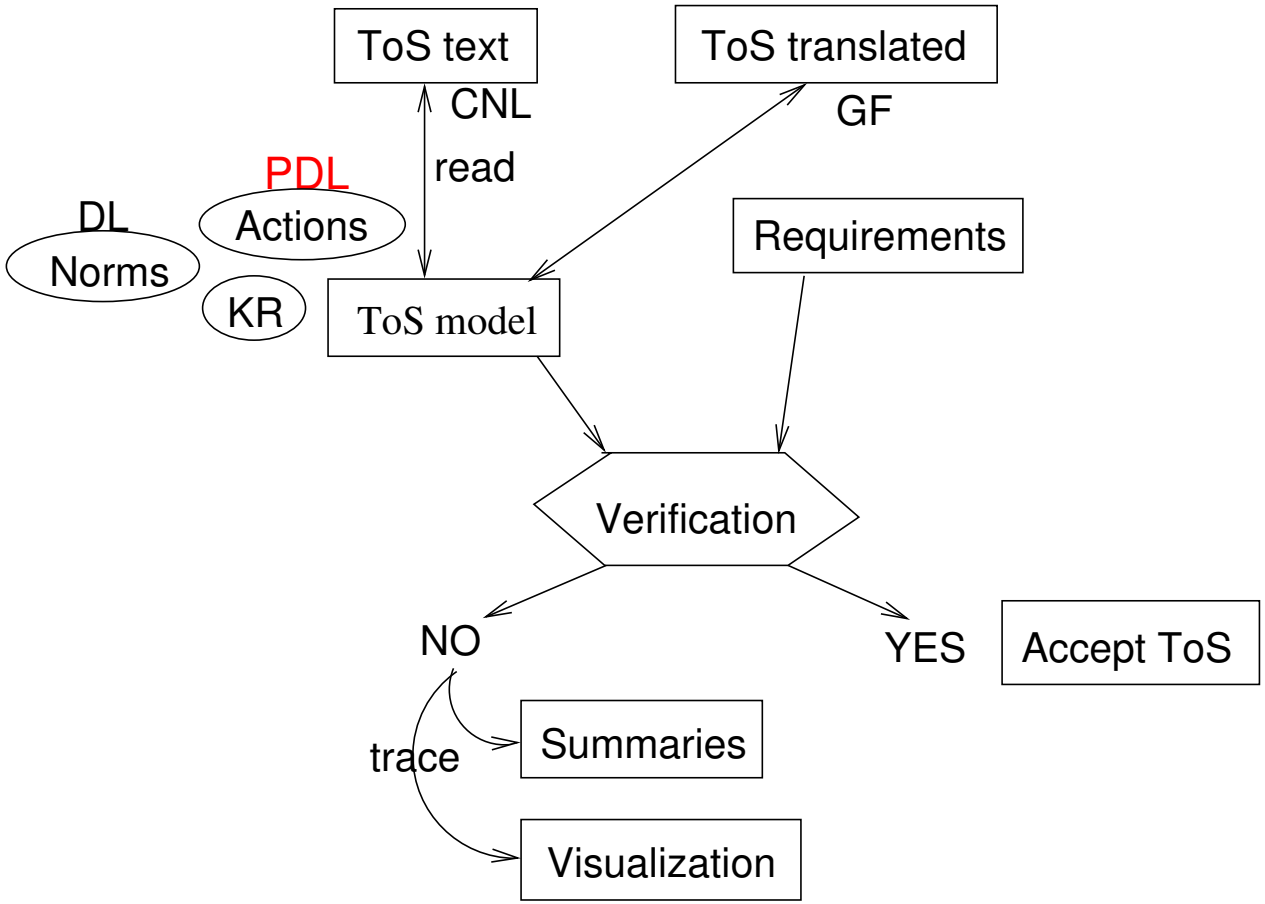
Technologies and Logics

where do they fit in LoToS ?



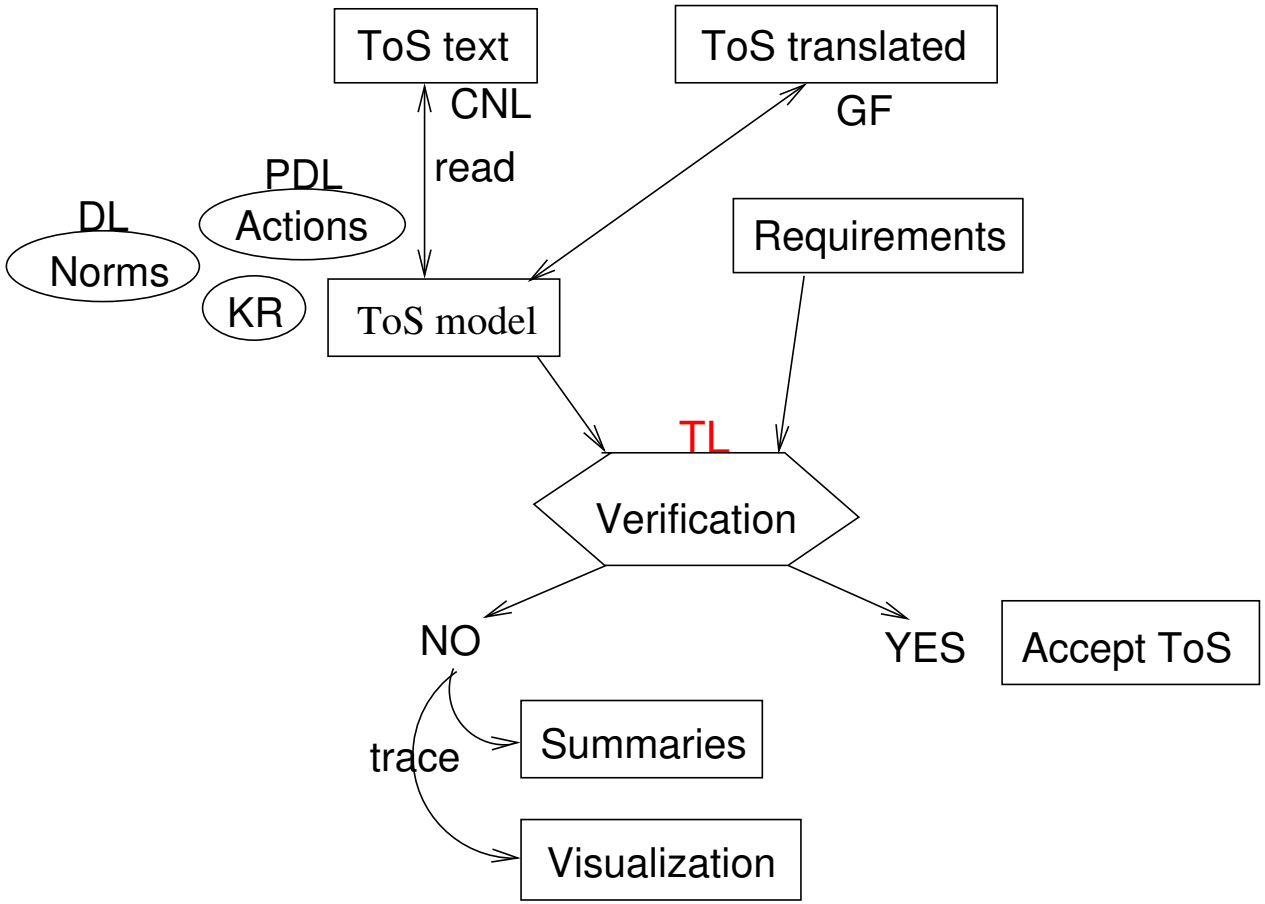
Technologies and Logics

where do they fit in LoToS ?



Technologies and Logics

where do they fit in LoToS ?



Automate the Reading

using Controlled Natural Languages.

Automate the Reading

using Controlled Natural Languages.

Attempto/ACE
from Zurich University
http://attempto.ifi.uzh.ch

Automate the Reading

using Controlled Natural Languages.

Attempto/ACE
from Zurich University
http://attempto.ifi.uzh.ch

based on Discourse Representation Theory [Book of H.Kamp&U.Reyle]

Automate the Reading

using Controlled Natural Languages.

Attempto/ACE
from Zurich University
http://attempto.ifi.uzh.ch

based on Discourse Representation Theory [Book of H.Kamp&U.Reyle]

– Mature, quite expressive, and with a wealth of tools around.
– Attempto Controlled English is a restricted natural language.
Not clear if ToS language fits ACE restrictions.
Work exists, e.g. from Stefan Höfler.

Automate the Reading using

Controlled Natural Languages (DRT/ACE)

Are using the expressiveness of First Order Logic (FOL).

- FOL is well studied and with a wealth of tools.
- Rule-based reasoning is related to FOL as Horn clauses.

Controlled languages are well known in areas like engineering.
(See IBM specifications.)

May be difficult to impose in Law.

- How to transition smooth from ToS texts to controlled Law language?
- How to combine CNL with Knowledge Representation?
- How to allow/handle ambiguities?

Automated Translation

Automated Translation

Grammatical Framework (GF)

from Chalmers University

http://www.grammaticalframework.org/

Automated Translation

Grammatical Framework (GF)
from Chalmers University
http://www.grammaticalframework.org/

based on Martin-Löf's intuitionistic type theory

– Mature and with growing community.
– Connection with Attempto controlled English.
– Functional style language.

Knowledge Representation

for Law

To capture relationships between the meaning of legal definitions and actions.

Knowledge Representation

for Law

To capture relationships between the meaning of legal definitions and actions.

- An ontology for the legal domain.
- Good experience exists in e.g., ontologies for medicine or biology
- OWL is a widespread language for building ontologies.

Knowledge Representation

for Law

To capture relationships between the meaning of legal definitions and actions.

- An ontology for the legal domain.
- Good experience exists in e.g., ontologies for medicine or biology
- OWL is a widespread language for building ontologies.

For the legal domain we may look at:

- The ESTRELLA project and the LKIF language
- CEN MetaLex open XML format for interchange of legal and legislative documents.
- Monitor the works of The Leibniz Center for Law in Amsterdam or the blog VOXPOPULII from Cornell University

Knowledge Representation – Ontologies

Ontologies are built using Description Logics

- Many variants of Description Logics exist
  – depending on expressive power
  – depending on computational complexity

- DL Lite has good computational complexity, used in medicine
- OWL is well adopted for semantic web, because of good expressiveness

- See Oxford group of Ian Horrocks
- The Description Logic Handbook

Normative notions

Standard notions like:
Rights/Permissions, Obligations, Prohibitions/Forbidden

Normative notions

Standard notions like:
Rights/Permissions, Obligations, Prohibitions/Forbidden

- In natural language:
  - “may”, “can”, “permitted”, “has the right”
  - “must”, “should”, “is obliged”, “is expected”
  - “not allowed”, “must not”

Normative notions

Standard notions like:
Rights/Permissions, Obligations, Prohibitions/Forbidden

- In natural language:
  - "may", "can", "permitted", "has the right"
  - "must", "should", "is obliged", "is expected"
  - "not allowed", "must not"

- See DeonticLogic.org
- The Handbook of Deontic Logic and Normative Systems
- or the DEON conferences

Normative notions

Standard notions like:
Rights/Permissions, Obligations, Prohibitions/Forbidden

- In natural language:
  - "may", "can", "permitted", "has the right"
  - "must", "should", "is obliged", "is expected"
  - "not allowed", "must not"

- See DeonticLogic.org
- The Handbook of Deontic Logic and Normative Systems
- or the DEON conferences

More notions like:
Powers, Governing policies, Exceptions, Parties/Roles, Delegation

- of importance to ToS
- but no satisfactory theories exist yet.

Normative notions and Actions

Actions abound in legal contracts (and in ToS)

Normative notions and Actions

Actions abound in legal contracts (and in ToS)

- Deontic modalities applied over actions;
  "Obligatory to pay rent", "Forbidden to download more than 5Mb"
- Actions may have complex structure, durations, quantities, or roles.

Normative notions and Actions

Actions abound in legal contracts (and in ToS)

- Deontic modalities applied over actions;
  "Obligatory to pay rent", "Forbidden to download more than 5Mb"
- Actions may have complex structure, durations, quantities, or roles.

Computer science studies many formalisms for actions:

- Propositional Dynamic Logic (PDL) used for regular expressions.
  [PDL Book by Harel&Kosen&Tiurin]
- Dynamic Deontic Logic describes deontic modalities over actions in the style of PDL. (see [J.-J. Ch. Meyer], [K. Segerberg])
- Process algebras are describing complex structured actions
  (see mCRL2 and tool set)

Normative notions and Temporal order

Temporal Logics reason about properties that change over time.

- Time is a linear order, and properties hold at time points
- Temporal operators capture notions like:
  - Property holds always in the future (or past)
  - or at some eventual future point
  - $Prop_1$ holds at all points until $Prop_2$ becomes true

Normative notions and Temporal order

Temporal Logics reason about properties that change over time.

- Time is a linear order, and properties hold at time points
- Temporal operators capture notions like:
  - Property holds always in the future (or past)
  - or at some eventual future point
  - $Prop_1$ holds at all points until $Prop_2$ becomes true
- Combinations of temporal operators with deontic logics and logics of actions have been investigated.
- Model checking is a technique well studied for temporal logics to check if a model satisfies a logic formula/property.

Verifying requirements on ToS

through Model Checking

Verifying requirements on ToS

through Model Checking

- Requirements are defined as a formula in an appropriate logic, depending on what the requirement is about.
- The ToS has been read into a model for this logic.

Verifying requirements on ToS

through Model Checking

- Requirements are defined as a formula in an appropriate logic, depending on what the requirement is about.
- The ToS has been read into a model for this logic.

Model checking is the technique that
checks a logical formula against a logical model.

- Model checking is automatic.
- Answers YES/NO (counter-example), in a Boolean setting
  or gives probabilistic answers

Verifying requirements on ToS

through Model Checking

- Requirements are defined as a formula in an appropriate logic, depending on what the requirement is about.
- The ToS has been read into a model for this logic.

Model checking is the technique that
checks a logical formula against a logical model.

- Model checking is automatic.
- Answers YES/NO (counter-example), in a Boolean setting
  or gives probabilistic answers
- Is computationally intensive
  (depending on the dimension of the model)

Negotiation

based on Verification

Negotiation

based on Verification

Negotiation can only happen before a ToS is accepted

Negotiation

based on Verification

Negotiation can only happen before a ToS is accepted
and only when the ToS fails to satisfy some of the user requirements

Negotiation

based on Verification

Negotiation can only happen before a ToS is accepted
and only when the ToS fails to satisfy some of the user requirements

- Use the explanation/counter-example/error-trace to change the ToS
- Send the satisfactory ToS back to the other party (service provider)
- Each party does the same verification-change-send until satisfied

Negotiation

based on Verification

Negotiation can only happen before a ToS is accepted
and only when the ToS fails to satisfy some of the user requirements

- Use the explanation/counter-example/error-trace to change the ToS
- Send the satisfactory ToS back to the other party (service provider)
- Each party does the same verification-change-send until satisfied

Problems?

- An automated negotiation could loop forever.

Negotiation

based on Verification

Negotiation can only happen before a ToS is accepted
and only when the ToS fails to satisfy some of the user requirements

- Use the explanation/counter-example/error-trace to change the ToS
- Send the satisfactory ToS back to the other party (service provider)
- Each party does the same verification-change-send until satisfied

Problems?

- An automated negotiation could loop forever.
  – A measure of redundancy (minimal change) must exist.
  – User intervention can stop the negotiation.

Negotiation

based on Verification

Negotiation can only happen before a ToS is accepted
and only when the ToS fails to satisfy some of the user requirements

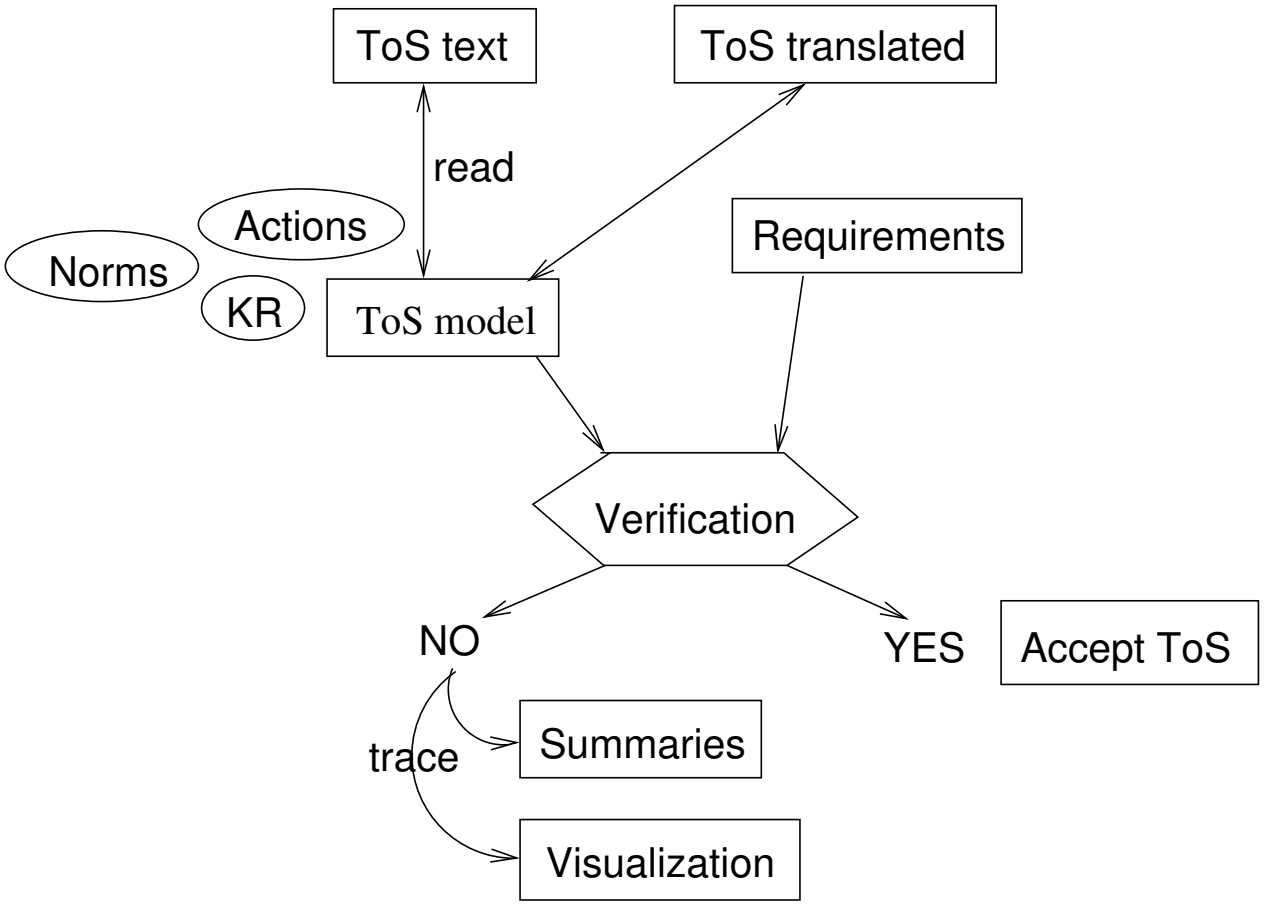
- Use the explanation/counter-example/error-trace to change the ToS
- Send the satisfactory ToS back to the other party (service provider)
- Each party does the same verification-change-send until satisfied

Problems?

- An automated negotiation could loop forever.
  – A measure of redundancy (minimal change) must exist.
  – User intervention can stop the negotiation.
- Simple negotiation parameters can effectively terminate;
  e.g.: involving quantifiable parameters s.a. deadlines or amounts,
  even requirements expressed as logical formulas
- How about privacy requirements?!

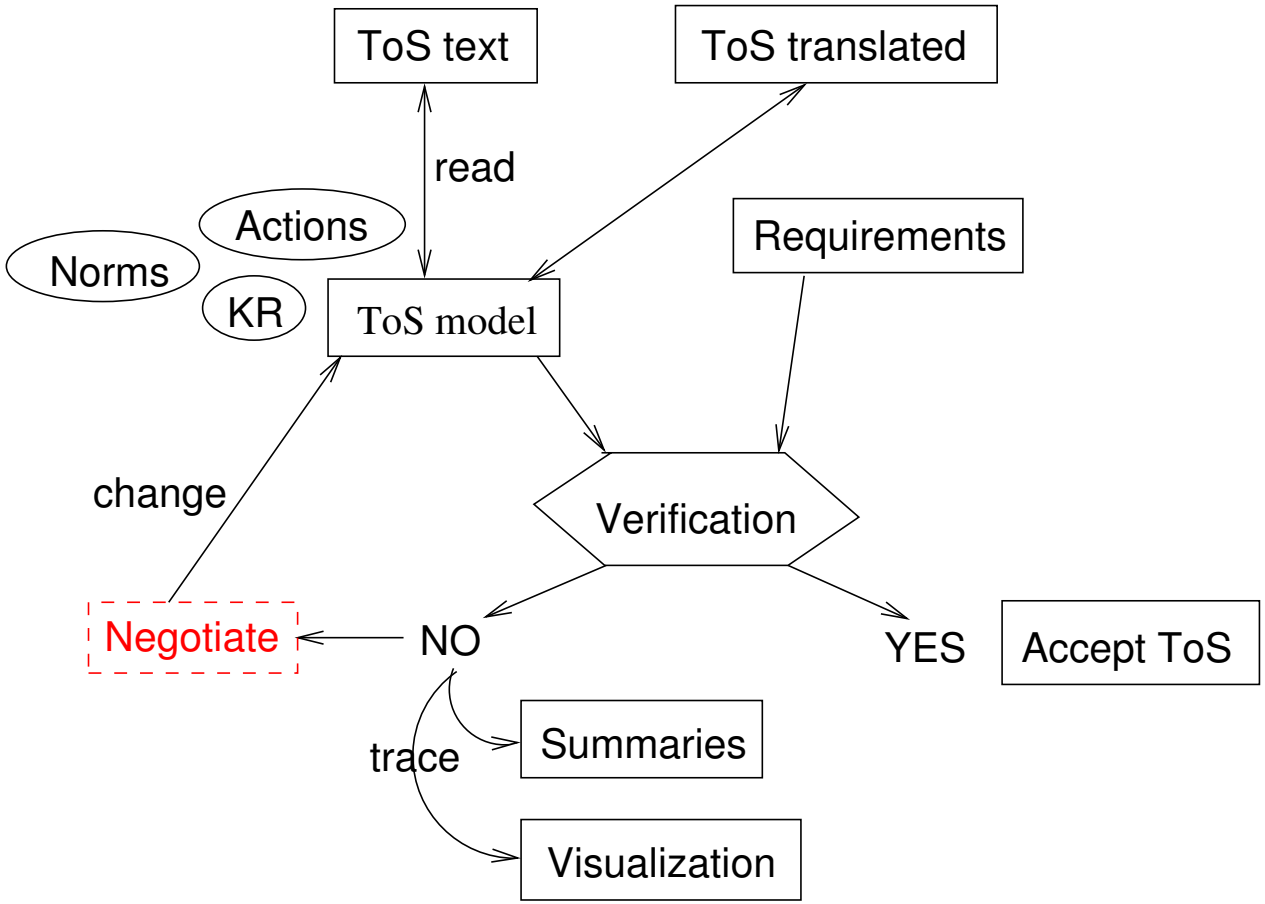
Negotiation

where does it fit in LoToS ?



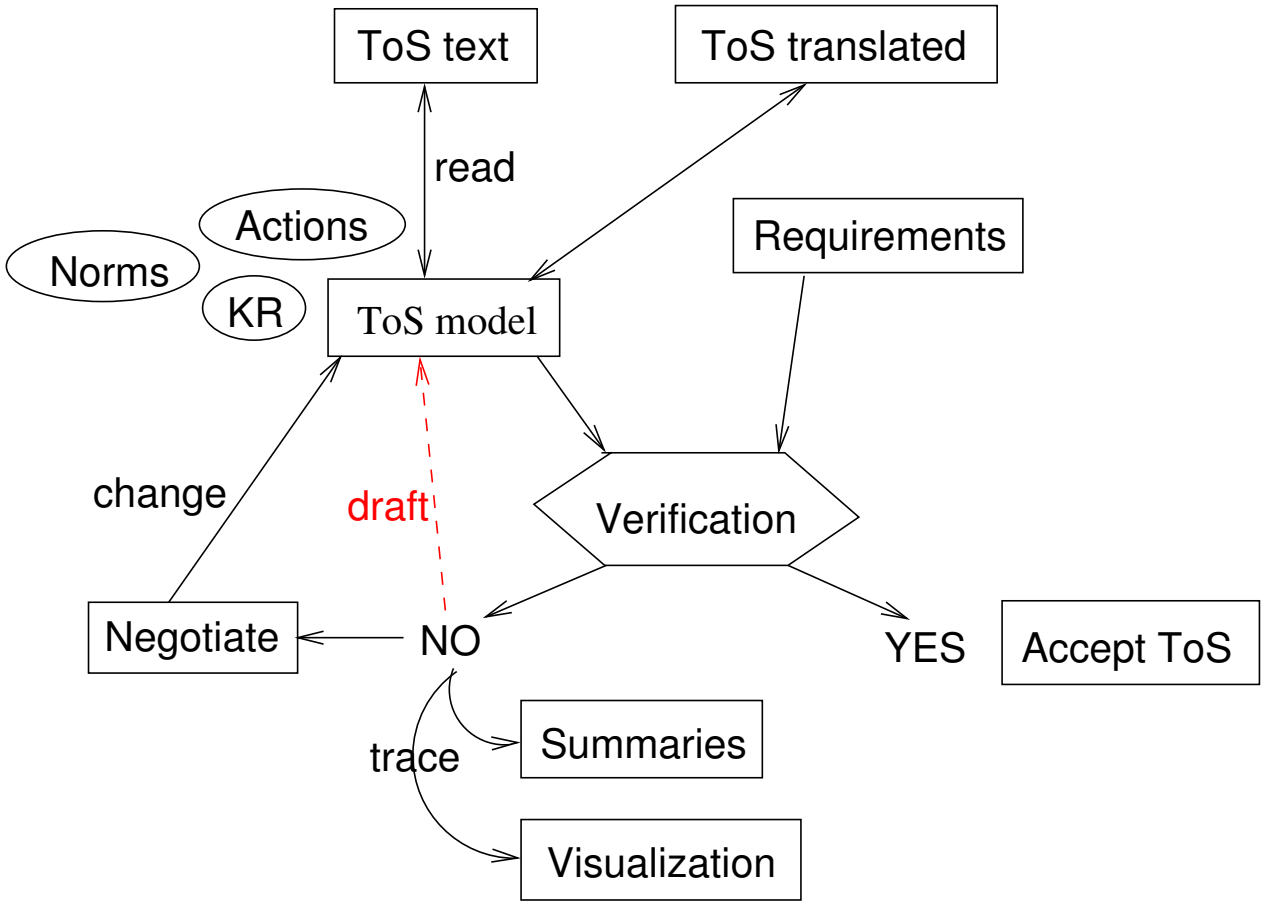
Negotiation

where does it fit in LoToS ?



Negotiation

where does it fit in LoToS ?



Monitoring

based on Verification

Monitoring can only be done after the ToS is accepted.

Monitoring

based on Verification

Monitoring can only be done after the ToS is accepted.

Uses:

- Describe sequence of actions and see if they conform with the ToS
- Quantitative evaluation of gains/losses of a sequence of actions wrt. the ToS even if not fully conformant
- Adapt technology from software monitoring

Monitoring

based on Verification

Monitoring can only be done after the ToS is accepted.

Uses:

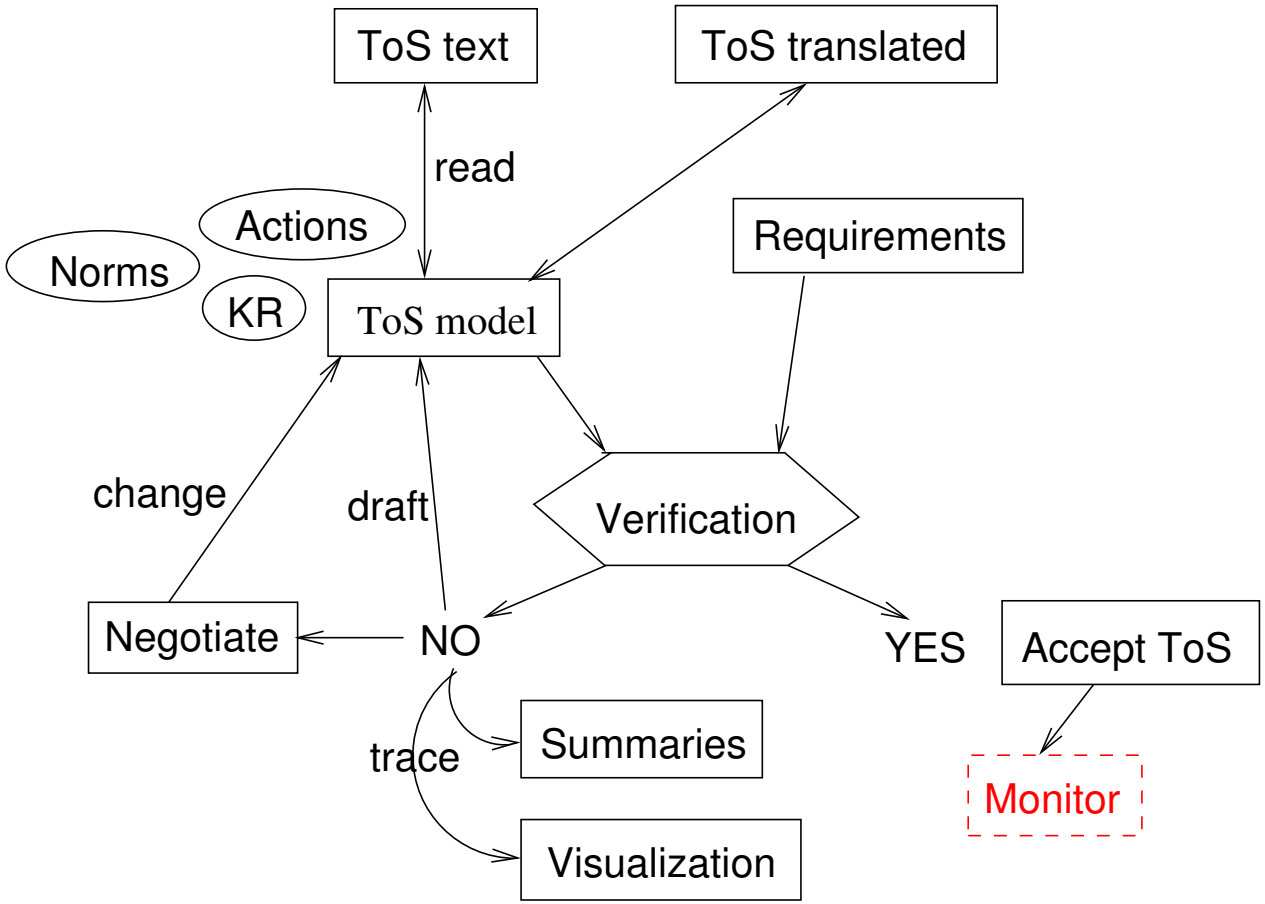
- Describe sequence of actions and see if they conform with the ToS
- Quantitative evaluation of gains/losses of a sequence of actions wrt. the ToS even if not fully conformant
- Adapt technology from software monitoring

For non-expert users:

- For the existing ToS in the on-line LoToS, predefined (non)acceptable sequence of actions can be searched
- or Templates of sequences of actions (also for parametric actions) visualization

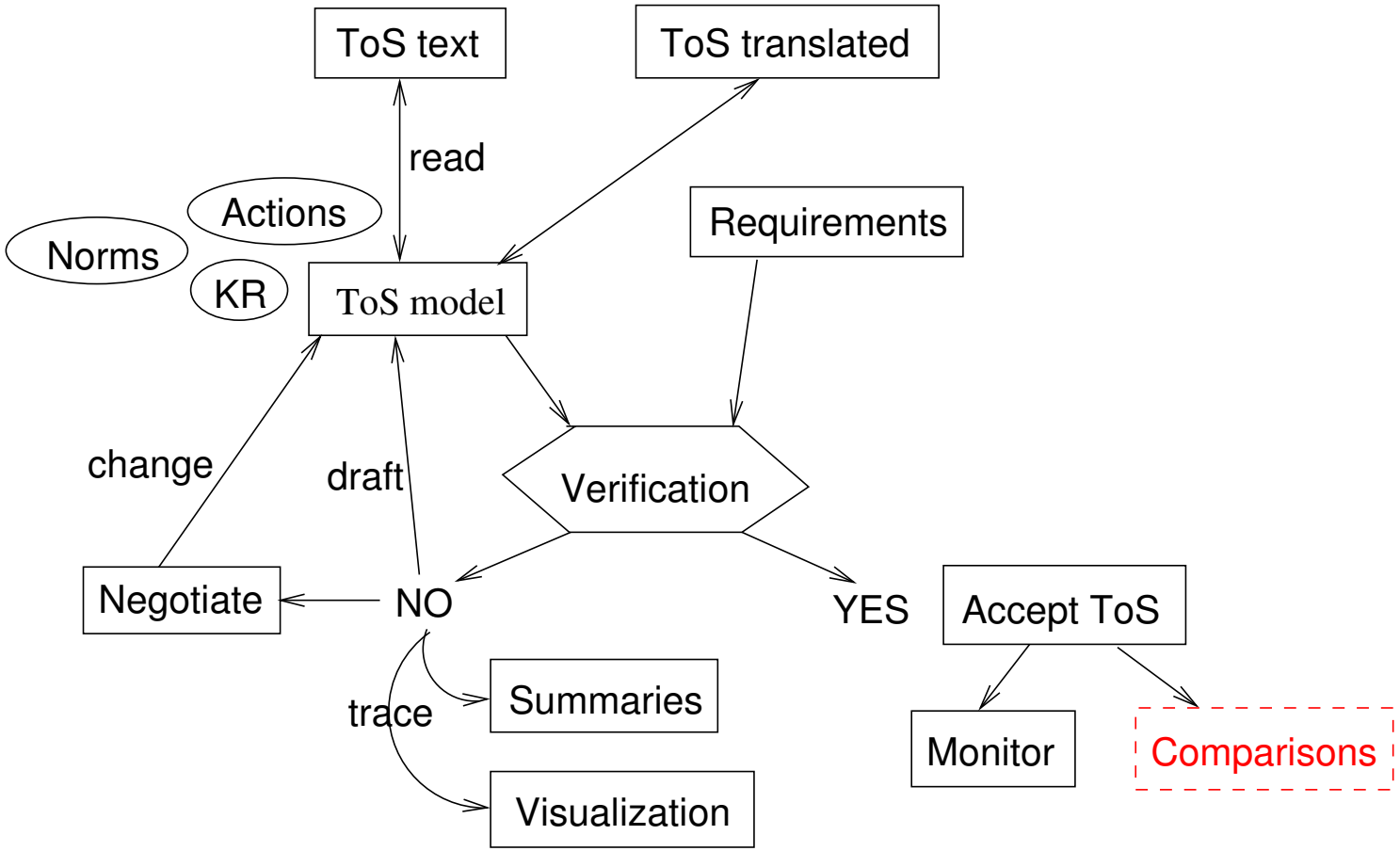
LoToS summary

diagrammatic



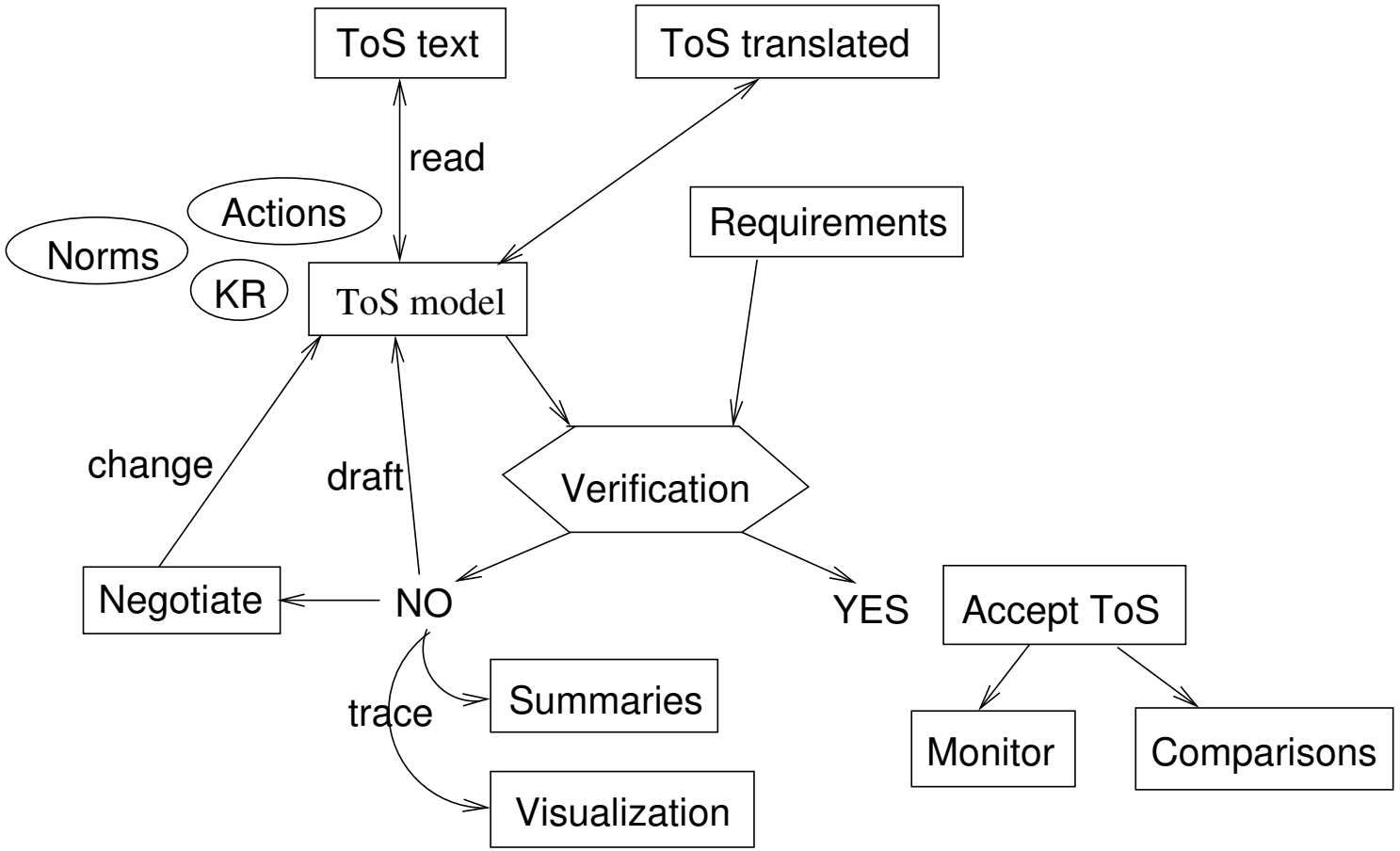
LoToS summary

diagrammatic



LoToS summary

diagrammatic

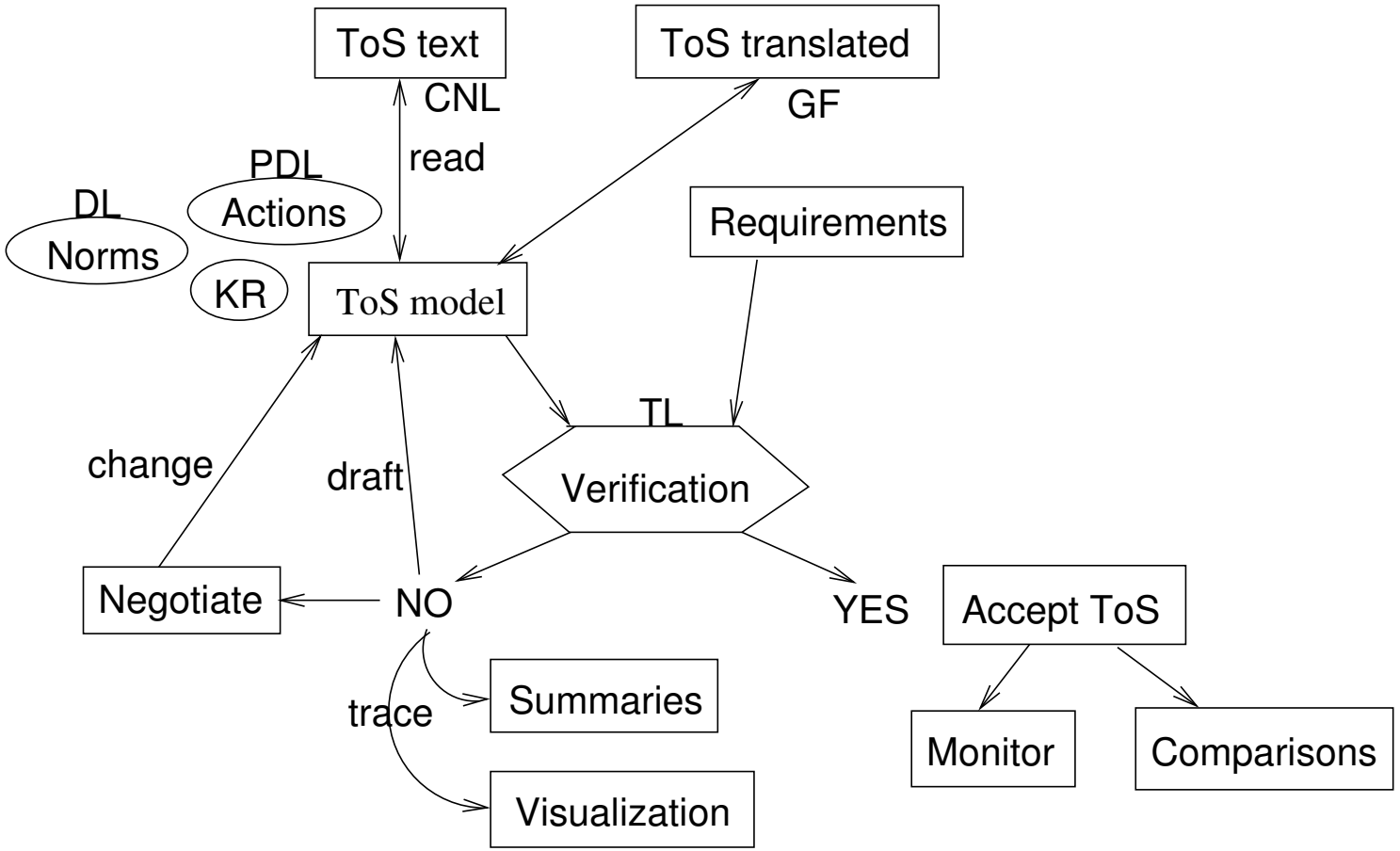


On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

LoToS summary

diagrammatic



On–line LoToS helps non–experts

Models $\models$ Requirements Templates Experts Computing power

Plan for the Workshop

- Discussions and comments pro/contra
- Answers and Proposals of solutions
- Testing/interacting with existing tools for the presented technologies.
- Share your related work. How/Where would you apply LoToS?
- What do you expect from LoToS? More questions from me ...

Plan for the Workshop

- Discussions and comments pro/contra
- Answers and Proposals of solutions
- Testing/interacting with existing tools for the presented technologies.
- Share your related work. How/Where would you apply LoToS?
- What do you expect from LoToS? More questions from me ...

Thank you for the attention!
Welcome after the break.