e-learning
Idea
CTP — tutoring
TSAC tutor

languages ?

TSAC's language

Language design
generalized ?

Convergent architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summar

Program Languages with CTP Features? On ISAC-experiments with Isabelle'09

Walther Neuper

Institute for Softwaretechnology Graz University of Technology

Workshop on Formal and Automated Theorem Proving January 2010, Beograd

Convergen architecture Isabelle history *TSAC* joins

Summa

Outline

- 1 Issues from e-learning
 Idea
 CTP tutoring

 ISAC tutor demonstration
- 2 CTP-based languages ?

 \(\mathcal{ISAC} \) s language

 Language design generalized ?
- 3 Convergent architecture Isabelle history \(\mathcal{TSAC} \) joins Isabelle
- 4 Summary

Idea

demonstration

ISAC's language generalized?

1 Issues from e-learning Idea

CTP — tutoring

- Isabelle history

Issues from e-learning

CTP — tutoring

TSAC tutor
demonstration

CTP-based languages?

TSAC's language
Language design
generalized?

Convergent architecture Isabelle history

TSAC* joins

Summar

Design a program language for applied mathematics . . .

Design a language analoguous to CAS-based languages

but based on Computer Theorem Proving (CTP) such that programs implementing applied math automatically create tutoring on that math stuff.

e-learning Idea

 $\begin{array}{c} \mathsf{CTP} - \mathsf{tutoring} \\ \mathcal{ISAC} \mathsf{\ tutor} \\ \mathsf{demonstration} \end{array}$

CTP-based languages?

TSAC's language
Language design generalized?

Convergent architecture Isabelle history

TSAC joins**

Summar

Design a program language for applied mathematics . . .

Design a language analoguous to CAS-based languages

but based on Computer Theorem Proving (CTP)

automatically create tutoring on that math stuff.

e-learning

CTP — tutoring

**TSAC* tutor
demonstration

CTP-based languages?

**TSAC's language Language design generalized?

Convergen architecture Isabelle history

USAC joins Isabelle

Summar

Design a program language for applied mathematics . . .

Design a language analoguous to CAS-based languages but based on Computer Theorem Proving (CTP) such that programs implementing applied math

e-learning Idea

CTP — tutoring

ISAC tutor

demonstration

CTP-based languages?

TSAC's language
Language design generalized?

Convergen architecture Isabelle history

USAC joins Isabelle

Summar

Design a program language for applied mathematics . . .

Design a language analoguous to CAS-based languages but based on Computer Theorem Proving (CTP) such that programs implementing applied math automatically create tutoring on that math stuff.

e-learning

CTP — tutoring

TSAC tutor

demonstration

CTP-based languages?

**TSAC's language Language design generalized?

Convergen architecture Isabelle history

USAC joins Isabelle

Summar

Design a program language for applied mathematics . . .

Design a language analoguous to CAS-based languages but based on Computer Theorem Proving (CTP) such that programs implementing applied math automatically create tutoring on that math stuff.

Summar

Outline

- 1 Issues from e-learning Idea CTP tutoring
 - ISAC tutor demonstration
- 2 CTP-based languages ?

 ISAC's language

 Language design generalized ?
- 3 Convergent architecture Isabelle history

 ISAC joins Isabelle**
- 4 Summary

e-learning

CTP — tutoring

CTP-based languages?

TSAC's language

Language design generalized ?

architecture
Isabelle history

\(\mathcal{I} \mathcal{S} \mathcal{A} \mathcal{C} \) joins

Summar

Requirements in tutoring applied math

A tutoring system for applied math serves by ...

- 1 checking user-input "correct modulo a theory"
- 2 providing surveys on subproblems and specifications
- guiding the user step-wise towards a solution

CTP-based languages ?

ISAC's language Language design generalized?

architecture
Isabelle history

TSAC joins

Summar

Requirements in tutoring applied math

A tutoring system for applied math serves by ...

- 1 checking user-input "correct modulo a theory"
 - providing surveys on subproblems and specifications
- guiding the user step-wise towards a solution

CTP — tutoring

CTP-based languages?

ISAC's language Language design generalized?

architecture
Isabelle history

TSAC joins
Isabelle

Summar

Requirements in tutoring applied math

A tutoring system for applied math serves by ...

- 1 checking user-input "correct modulo a theory"
- providing surveys on subproblems and specifications
- 3 guiding the user step-wise towards a solution

Convergent architecture Isabelle history

\[\mathcal{ISAC} \]

JSAC joins

Summar

Requirements in tutoring applied math

A tutoring system for applied math serves by ...

- 1 checking user-input "correct modulo a theory"
- providing surveys on subproblems and specifications
- guiding the user step-wise towards a solution

e-learning

Idea

CTP — tutoring

CTP-based languages?

TSAC's language

Convergent architecture Isabelle history \(\mathcal{LSAC} \) joins Isabelle

generalized?

Summar

Requirements in tutoring applied math

A tutoring system for applied math serves by ...

- 1 checking user-input "correct modulo a theory"
- providing surveys on subproblems and specifications
- guiding the user step-wise towards a solution

e-learning

Idea

CTP — tutoring

TSAC tutor

demonstration

CTP-based languages?

TSAC's language
Language design generalized?

Convergent architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summar

1 Issues from e-learning
Idea
CTP — tutoring
TSAC tutor demonstration

- 3 Convergent architecture Isabelle history

 TSAC joins Isabelle**
- 4 Summary

Convergent architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summar

Resume of the demonstration

The \mathcal{ISAC} tutor serves with . . .

- 1 checking user-input "correct modulo a theory" by use of Isabelle provers (e.g. simplifier): CTP
- 2 providing surveys on subproblems and specifications by use of Isabelle contexts (e.g. pre-conditions): CTP
- 3 guiding the user step-wise towards a solution using a single-stepping interpreter: program language

Convergen architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summar

Resume of the demonstration

The \mathcal{ISAC} tutor serves with . . .

- 1 checking user-input "correct modulo a theory" by use of Isabelle provers (e.g. simplifier): CTP |
- 2 providing surveys on subproblems and specifications by use of Isabelle contexts (e.g. pre-conditions): CTP
- 3 guiding the user step-wise towards a solution using a single-stepping interpreter: program language

Summar

Resume of the demonstration

The \mathcal{ISAC} tutor serves with . . .

- 1 checking user-input "correct modulo a theory" by use of Isabelle provers (e.g. simplifier): CTP |
- 2 providing surveys on subproblems and specifications by use of Isabelle contexts (e.g. pre-conditions): CTP |
- 3 guiding the user step-wise towards a solution using a single-stepping interpreter: program language

Convergent architecture Isabelle history

\[\mathcal{TSAC} \] joins Isabelle

Summar

Resume of the demonstration

The \mathcal{ISAC} tutor serves with . . .

- 1 checking user-input "correct modulo a theory" by use of Isabelle provers (e.g. simplifier): CTP /
- 2 providing surveys on subproblems and specifications by use of Isabelle contexts (e.g. pre-conditions): CTP |
- 3 guiding the user step-wise towards a solution using a single-stepping interpreter: program language

Convergent architecture Isabelle history

\[\mathcal{TSAC} \] joins Isabelle

Summar

Resume of the demonstration

The \mathcal{ISAC} tutor serves with . . .

- 1 checking user-input "correct modulo a theory" by use of Isabelle provers (e.g. simplifier): CTP!
- 2 providing surveys on subproblems and specifications by use of Isabelle contexts (e.g. pre-conditions): CTP!
- 3 guiding the user step-wise towards a solution using a single-stepping interpreter: program language !

Convergent architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summar

Resume of the demonstration

The \mathcal{ISAC} tutor serves with . . .

- 1 checking user-input "correct modulo a theory" by use of Isabelle provers (e.g. simplifier): CTP!
- 2 providing surveys on subproblems and specifications by use of Isabelle contexts (e.g. pre-conditions): CTP!
- guiding the user step-wise towards a solution using a single-stepping interpreter: program language!

languages?

ISAC's language Language design generalized?

architecture
Isabelle history

TSAC joins
Isabelle

Summar

- 1 Issues from e-learning
 Idea
 CTP tutoring

 ISAC tutor demonstration
- 2 CTP-based languages ?

 **TSAC's language*

 Language design generalized ?
- 3 Convergent architecture Isabelle history

 ISAC joins Isabelle**
- 4 Summary

e-learning
Idea

CTP — tutoring

ISAC tutor

CTP-based languages?

TSAC's language Language design generalized?

Convergen architecture Isabelle history

Summar

Resume of the demonstration

\mathcal{ISAC} 's experimental program language . . .

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation
- 3 maintans contexts (predicates, type-constraints) which assists in checking user-input; programming: logic checks in runtime improve safety!

Summar

Resume of the demonstration

 \mathcal{ISAC} 's experimental program language ...

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation
- 3 maintans contexts (predicates, type-constraints) which assists in checking user-input; programming: logic checks in runtime improve safety!

CTP-based languages?

TSAC's language Language design generalized?

Convergent architecture Isabelle history \mathcal{ISAC} joins

Summar

Resume of the demonstration

 \mathcal{ISAC} 's experimental program language ...

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation
- maintans contexts (predicates, type-constraints)
 which assists in checking user-input;
 programming: logic checks in runtime improve safety

architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summar

Resume of the demonstration

 \mathcal{ISAC} 's experimental program language ...

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- 2 checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation is
- maintans contexts (predicates, type-constraints)
 which assists in checking user-input;
 programming: logic checks in runtime improve safety!

e-learning
Idea

CTP — tutorin

ISAC tutor

CTP-based languages?

TSAC's language
Language design
generalized?

Convergen architecture Isabelle history \mathcal{ISAC} joins

Summar

Resume of the demonstration

 \mathcal{ISAC} 's experimental program language ...

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- 2 checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation!
- maintans contexts (predicates, type-constraints)
 which assists in checking user-input;
 programming: logic checks in runtime improve safety.

CTP-based languages?

Ianguages ?

TSAC's language

Language design
generalized ?

Convergent architecture Isabelle history

Summar

Resume of the demonstration

 \mathcal{ISAC} 's experimental program language ...

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- 2 checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation!
- 3 maintans contexts (predicates, type-constraints) which assists in checking user-input;

programming: logic checks in runtime improve safety!

Resume of the demonstration

 \mathcal{ISAC} 's experimental program language ...

- 1 is purely functional, user-in/output handled by interpreter; programming math in a typed, functional language!
- 2 checks specifications of subproblems, interactive specification is invoked by interpreter; programming: pre-conditions guard method invocation!
- 3 maintans contexts (predicates, type-constraints) which assists in checking user-input; programming: logic checks in runtime improve safety!

e-learning
Idea
CTP — tutoring
ISAC tutor
demonstration

CTP-based languages?

TSAC's language

Language design generalized ?

architecture
Isabelle history

TSAC joins** Isabelle

Summar

- 1 Issues from e-learning
 Idea
 CTP tutoring

 ISAC tutor demonstration
- 2 CTP-based languages ?

 \[\mathcal{ISAC} \cdots \text{ language} \\
 \text{Language design generalized ?} \]
- 3 Convergent architecture Isabelle history \(\mathcal{TSAC} \) joins Isabelle
- 4 Summary

e-learning

Idea

CTP — tutoring

ISAC tutor

CTP-based languages ?

TSA C's language Language design generalized ?

architecture
Isabelle history

TSAC joins
Isabelle

Summary

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 ... checks specifications of subproblems
- maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- 5 ... supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- 7 ...

e-learning
Idea
CTP — tutoring
ISAC tutor

CTP-based languages?

TSAC's language
Language design

generalized?

architecture
Isabelle history

TSAC joins
Isabelle

Summar

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 ... checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- Output
 <p
- supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- 7 ..

e-learning
Idea
CTP — tutoring
TSAC tutor

Convergent architecture Isabelle history \(\mathcal{ISAC} \) joins

Summar

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 ... checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- **5** ... supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- **7** ...

e-learning
Idea
CTP — tutoring
ISAC tutor
demonstration

Convergent architecture Isabelle history

Summary

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 ... checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- 5 ... supports proof of correctness of programs
- **6** ... supports local pretty printing (LATEX, MathML)
- **7** ...

e-learning
Idea
CTP — tutoring
TSAC tutor
demonstration

Convergen
architecture
Isabelle history

TSAC joins
Isabelle

Summar

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 . . . checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- **6** ... supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- 7 ...

e-learning
Idea
CTP — tutoring
TSAC tutor
demonstration

Convergen architecture Isabelle history

TSAC joins Isabelle

Summar

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 . . . checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- **6** ... supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- 7 ...

e-learning
Idea
CTP — tutoring
TSAC tutor
demonstration

CTP-based languages?

TSAC's language
Language design generalized?

Summar

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 . . . checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- **6** ... supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- 7 ..

... is this interesting for (appl.) math programmers?

To which other developments user guidance can hook up?

Summar

Features for CTP-based languages ?

A CTP-based language for (applied) math, which ...

- 1 ... is purely functional
- 2 ... checks specifications of subproblems
- 3 ... maintans contexts (predicates, type-constraints)
- 4 ... organizes knowledge local to theories, contexts
- **6** ... supports proof of correctness of programs
- 6 ... supports local pretty printing (LATEX, MathML)
- 7 ...

... is this interesting for (appl.) math programmers?

To which other developments user guidance can hook up?

Convergent architecture Isabelle history

generalized?

Summar

Outline

- Issues from e-learning
 Idea
 CTP tutoring
 ISAC tutor demonstration
- 2 CTP-based languages ?

 **ISAC's language

 Language design generalized ?
- 3 Convergent architecture Isabelle history ISAC joins Isabelle
- 4 Summary

Issues from e-learning

Idea

CTP — tutoring

TSAC tutor demonstration

CTP-based languages ?

Ianguages ?

ISAC's language
Language design
generalized ?

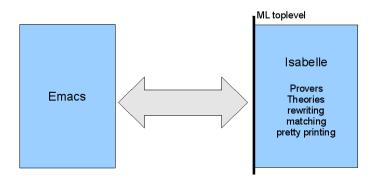
Convergent

Isabelle history

ISAC joins

Summary

The original Isabelle architecture



e-learning

 $\begin{array}{c} \mathsf{CTP} - \mathsf{tutoring} \\ \mathcal{ISAC} \mathsf{\ tutor} \\ \mathsf{demonstration} \end{array}$

CTP-based languages?

ISAC's language

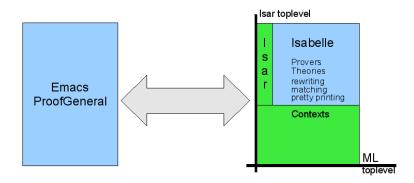
Language design generalized?

architecture

Isabelle history $\mathcal{ISAC} \text{ joins}$

Summary

Isar proof language hides goal/subgoal mechanism



e-learning

CTP — tutoring

TSAC tutor
demonstration

CTP-based languages?

Ianguages ?

ISAC's language

Language design

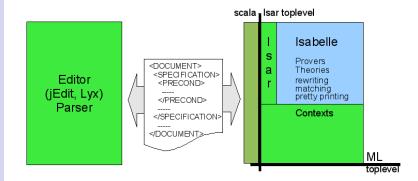
generalized ?

Convergen

Isabelle history $\mathcal{ISAC} \text{ joins}$

Summary

Scala will enhance interoperability for GUIs etc



CTP-based languages?

TSAC's language
Language design generalized?

Convergent architectur Isabelle history

TSAC joins Isabelle**

Summar

- 1 Issues from e-learning
 Idea
 CTP tutoring

 ISAC tutor demonstration
- 2 CTP-based languages ?

 **ISAC's language*

 Language design generalized ?
- 3 Convergent architecture Isabelle history \(\mathcal{ISAC} \) joins Isabelle
- 4 Summary

e-learning

demonstration
CTP-based

languages ?

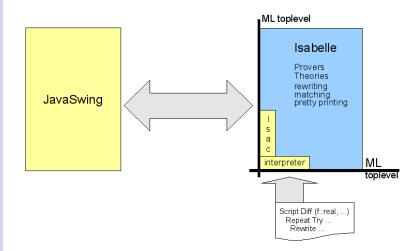
TSAC's language

Language design
generalized ?

Convergent architecture Isabelle history TSAC joins

Isabelle Summary

Present \mathcal{ISAC} adds Scripts and interpreter



e-learning
Idea
CTP — tutoring

demonstration

CTP-based languages?

languages?

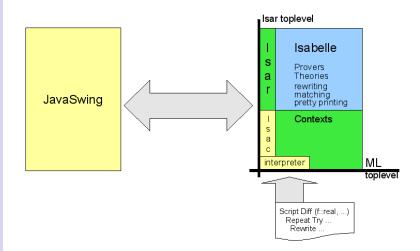
ISAC's language

Language design
generalized?

Convergent architecture Isabelle history \mathcal{ISAC} joins

Isabelle

The next version will exploit lsars contexts



e-learning

CTP — tutoring

TSAC tutor

demonstration

CTP-based languages?

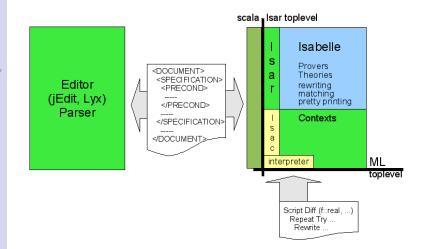
TSAC's language
Language design
generalized?

Convergent architecture

 \mathcal{ISAC} joins Isabelle

Summarv

$\mathcal{IS\!A\!C}$ adds programs (Isabelle terms, "Scripts") and interpreter



e-learning

CTP — tutoring

TSAC tutor

demonstration

CTP-based languages?

languages?

\(\mathcal{ISAC} \) s language

Language design
generalized?

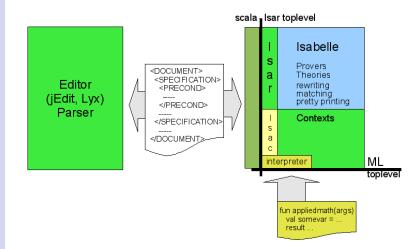
Convergen

Isabelle history

ISAC joins
Isabelle

Summarv

Could there be Standard ML instead the Isabelle terms?



e-learning

CTP — tutoring

TSAC tutor
demonstration

CTP-based languages?

Ianguages?

ISAC's language

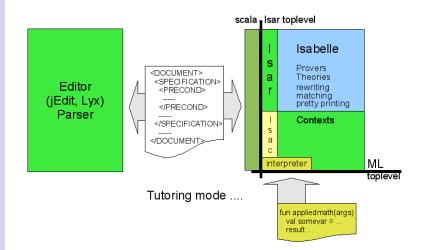
Language design
generalized?

Convergent architecture

 \mathcal{ISAC} joins Isabelle

Summary

Could there be Standard ML instead the Isabelle terms?



e-learning
Idea

demonstration

languages ?

\[\mathcal{ISAC} \) S language

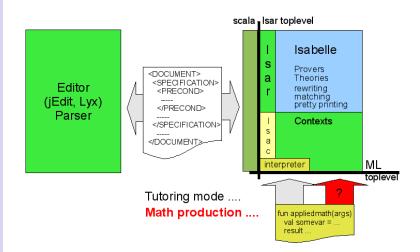
Language design
generalized ?

Convergen

 \mathcal{ISAC} joins Isabelle

Summary

Then the same program could be production code !?!



e-learning
Idea

CTP — tutoring

TSAC tutor

demonstration

CTP-based languages?

ISAC's language

generalized?

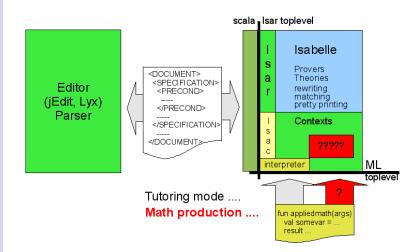
Convergentarchitecture

Isabelle history

ISAC joins
Isabelle

Summary

Then the same program could be production code !?!



e-learning

Idea

CTP — tutoring

CTP-based languages?

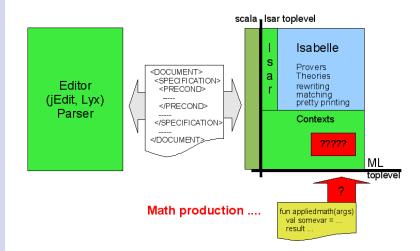
TSAC's language Language design generalized?

architecture Isabelle history

TSAC joins Isabelle

Summary

Then the same program could be production code !?!



Summary

Towards CTP-based languages ?

To features of CAS-based languages . . .

- typed matching and rewriting
- ... adding CTP-based ones towards a language, which...
 - ? is purely functional
 - ? checks specifications of subproblems
 - ? maintans contexts (predicates, type-constraints)
 - ? organizes knowledge local to theories, contexts
 - ? supports proof of correctness of programs
 - ? supports local pretty printing (LATEX, MathML)
 - ? etc ???

Ianguages ?

ISAC's language

Language design
generalized ?

Convergent architecture Isabelle history ISAC joins

Summary

Towards CTP-based languages ?

To features of CAS-based languages . . .

- typed matching and rewriting
- ... adding CTP-based ones towards a language, which...
 - ? is purely functional
 - ? checks specifications of subproblems
 - ? maintans contexts (predicates, type-constraints)
 - ? organizes knowledge local to theories, contexts
 - ? supports proof of correctness of programs
 - ? supports local pretty printing (LATEX, MathML)
 - ? etc ???

Language design generalized ?

Convergent architecture Isabelle history \(\mathcal{TSAC} \) joins Isabelle

Summary

Towards CTP-based languages ?

To features of CAS-based languages . . .

- typed matching and rewriting
- ... adding CTP-based ones towards a language, which...
 - ? is purely functional
 - ? checks specifications of subproblems
 - ? maintans contexts (predicates, type-constraints)
 - ? organizes knowledge local to theories, contexts
 - ? supports proof of correctness of programs
 - ? supports local pretty printing (LATEX, MathML)
 - ? etc ???

e-learning
Idea
CTP — tutoring

CTP-based languages ?

generalized?

Convergent architecture

architecture
Isabelle history

TSAC joins
Isabelle

Summary

Towards CTP-based languages ?

To features of CAS-based languages . . .

- typed matching and rewriting
- ... adding CTP-based ones towards a language, which...
 - ? is purely functional
 - ? checks specifications of subproblems
 - ? maintans contexts (predicates, type-constraints)
 - ? organizes knowledge local to theories, contexts
 - ? supports proof of correctness of programs
 - ? supports local pretty printing (LATEX, MathML)
 - ? etc ???

e-learning

Idea

CTP — tutoring

TSAC tutor

CTP-based languages?

TSAC's language
Language design generalized?

Convergent architecture Isabelle history \mathcal{ISAC} joins Isabelle

Summary

Towards CTP-based languages ?

To features of CAS-based languages . . .

- typed matching and rewriting
- ... adding CTP-based ones towards a language, which...
 - ? is purely functional
 - ? checks specifications of subproblems
 - ? maintans contexts (predicates, type-constraints)
 - ? organizes knowledge local to theories, contexts
 - ? supports proof of correctness of programs
 - ? supports local pretty printing (LTEX, MathML)
 - ? etc ???

CTP-based languages?

TSAC's language
Language design generalized?

Convergent architecture Isabelle history \mathcal{TSAC} joins Isabelle

Summary

Towards CTP-based languages ?

To features of CAS-based languages . . .

- typed matching and rewriting
- ... adding CTP-based ones towards a language, which...
 - ? is purely functional
 - ? checks specifications of subproblems
 - ? maintans contexts (predicates, type-constraints)
 - ? organizes knowledge local to theories, contexts
 - ? supports proof of correctness of programs
 - ? supports local pretty printing (LATEX, MathML)
 - ? etc ???