

PGIP as a prover-independent proof language

David Aspinall · Christoph Lüth



Introducing PGIP

The **Proof General Interaction Protocol**:

- A **protocol** for **interactive** provers to **interact** with a **user interface**.
 - **Proof General** reloaded
 - **Standard protocol** instead of **customised configuration**
- A generic **system architecture** to **connect** provers to **interfaces**

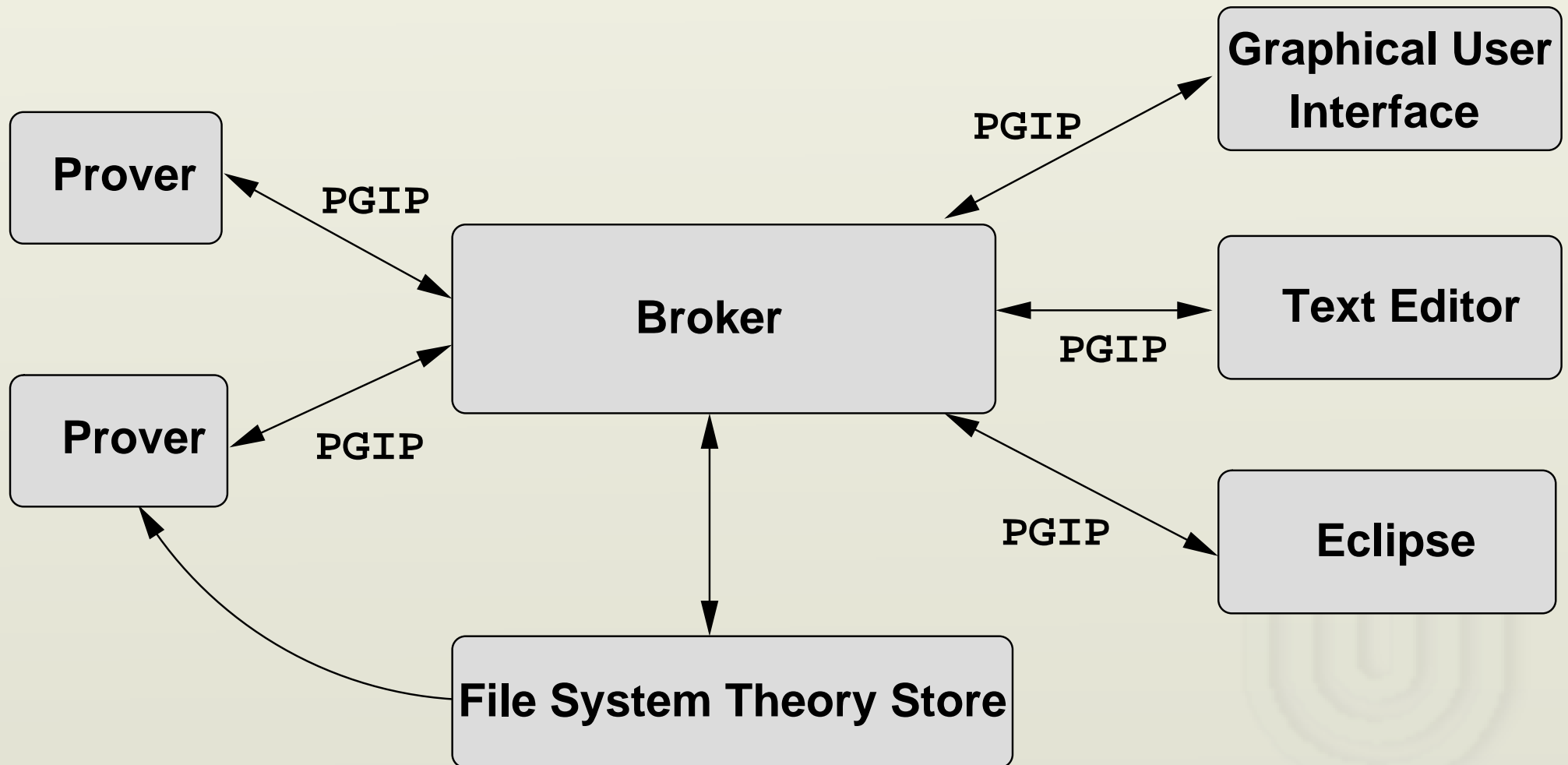
↪ a **Software Framework** for **Interactive Proof**

- **PG Kit**: reference implementation

The PGIP System Architecture

Prover Components

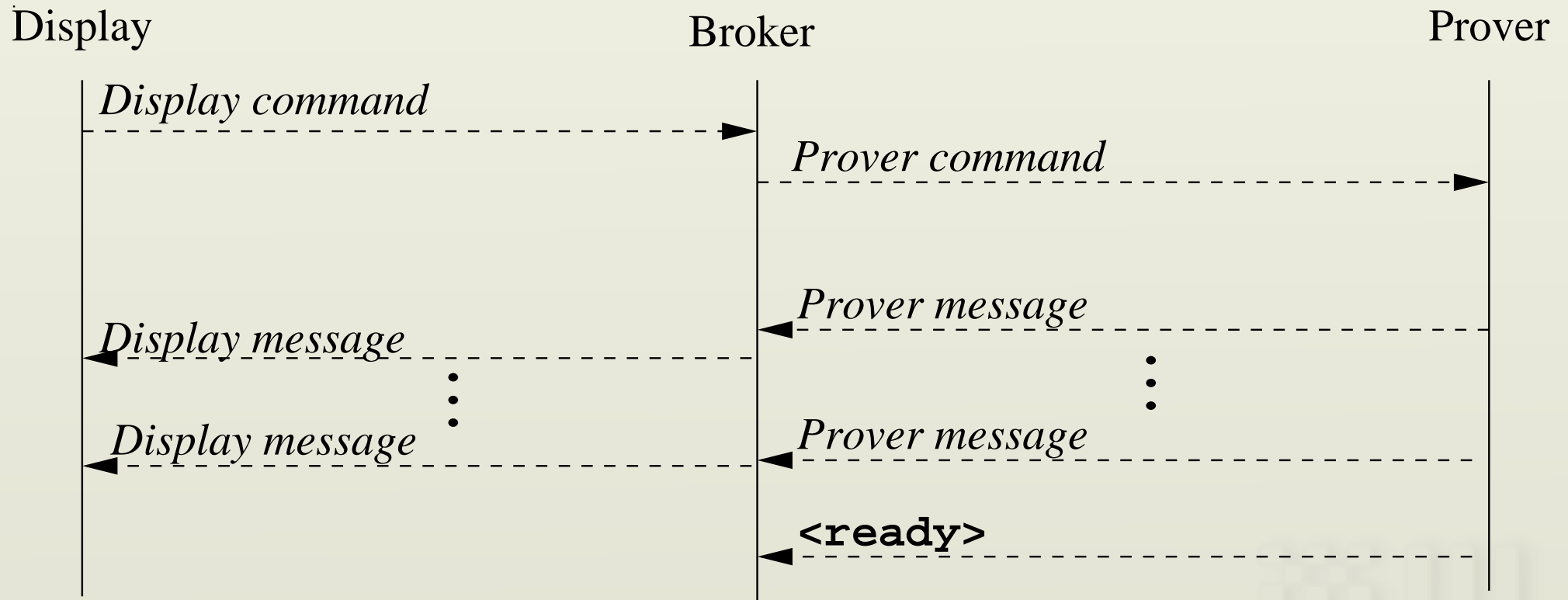
Display Components



PGIP Structure & Design Rationales

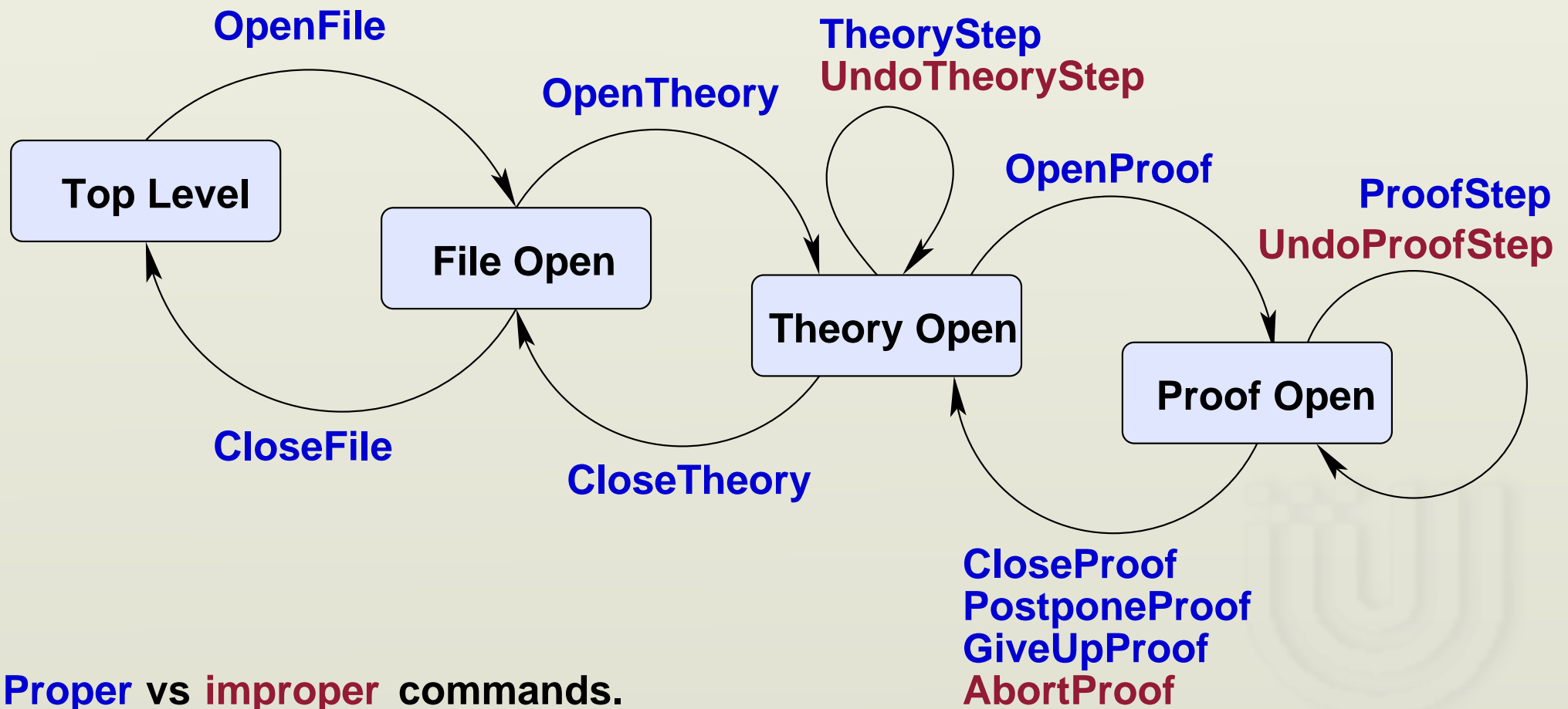
- PGIP **prover protocol**:
 - simple, small and powerful
 - easy to implement for existing provers
- PGIP **display protocol**:
 - nearly stateless, very verbose
 - easy to add new displays
- Cleverness in the central part — the **broker**
- **Ultimate aim**: replace **extra-logical** functionality; support **proof engineering**

PGIP Message Exchanges



PGIP Prover Model

- Abstract model of incremental proof development.



PGIP Script Model

Proof scripts: native language plus PGIP markup



PGIP Script Model

Proof scripts: native language plus PGIP markup

lemma fn1: "(EX x. P (f x)) \longrightarrow (EX y. P y)"

proof

 assume "EX x. P (f x)"

 thus "EX y. P y"

 proof

 fix a

 assume "P (f a)"

 show ?thesis ..

 qed

qed



PGIP Script Model

Proof scripts: native language plus PGIP markup

```
<opengoal name="fn1" >lemma fn1: &quot;(EX x. P (f x)) <sym name="lo
<openblock/><proofstep>proof</proofstep>
  <proofstep>assume &quot;EX x. P (f x)&quot;</proofstep>
  <proofstep>thus &quot;EX y. P y&quot;</proofstep>
<openblock/><proofstep>proof</proofstep>
  <proofstep>fix a</proofstep>
  <proofstep>assume &quot;P (f a)&quot;</proofstep>
  <proofstep>show ?thesis</proofstep><openblock/>
    <proofstep>..</proofstep><closeblock/>
  <proofstep>qed</proofstep><closeblock/>
<closegoal>qed</closegoal><closeblock/>
```

PGIP Script Model and Display Protocol

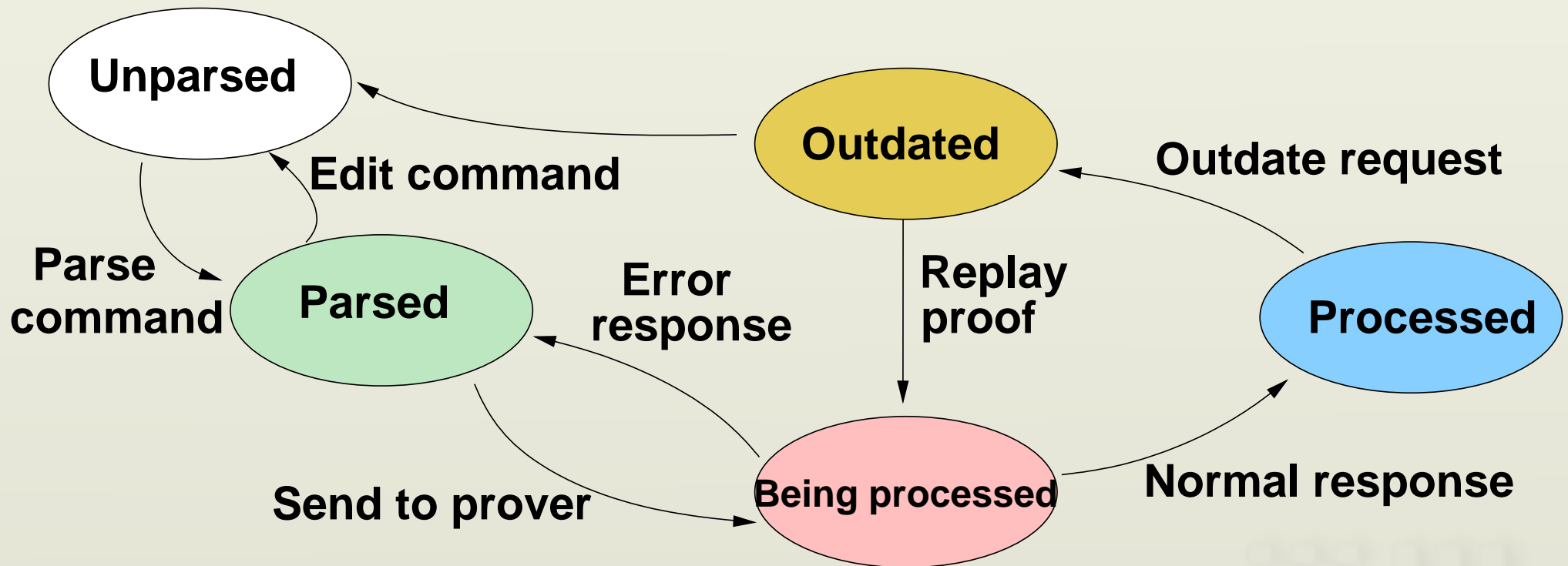
- **Proof scripts**: native language plus PGIP markup
- Markup makes **implicit structure explicit**
- Scripts split into **commands**



PGIP Script Model and Display Protocol

- **Proof scripts**: native language plus PGIP markup
- Markup makes **implicit structure explicit**
- Scripts split into **commands**
- Each **command** has a **state**
- **Proving** = processing commands = command state change

The Edit-Parse-Prove Cycle



The Broker

Central **Middleware** Component

- Handles **communication** and **setup**
- Maintains **proof scripts**
- Keeps track of **prover state**
- Converts **state-change requests** into **sequences of prover commands**



PGIP as a Prover-Independent Language

Theory T1

Definition 1

Lemma 1

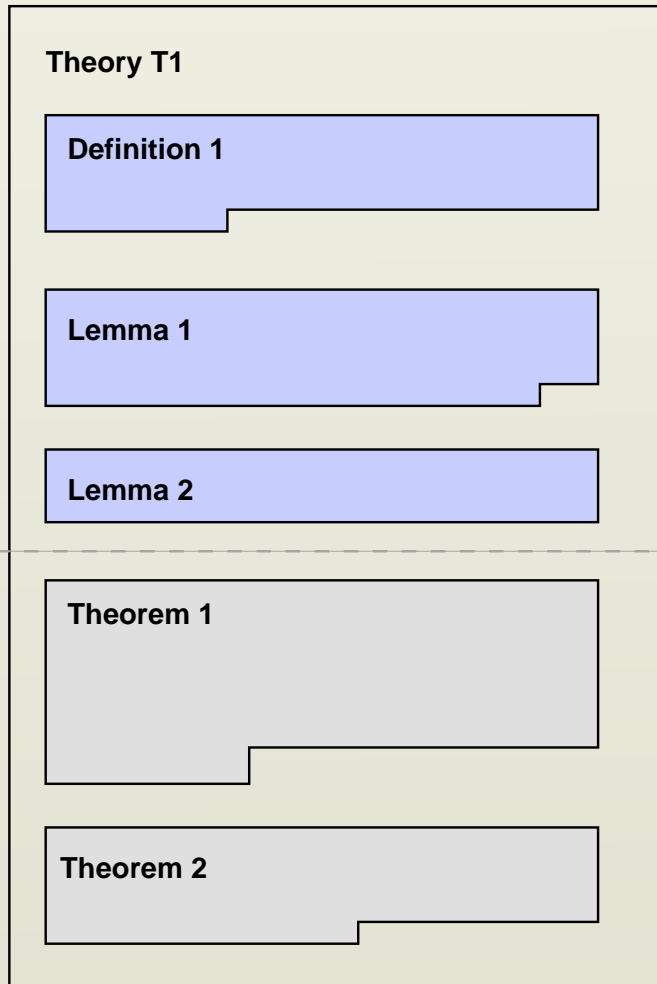
Lemma 2

Theorem 1

Theorem 2

- No assumptions about semantics
- Dependency management:
 - Guarantee reconstructibility
(synchronisation display \longleftrightarrow prover)
 - Change management
 - Linear vs. explicit dependencies

Theorem Dependencies



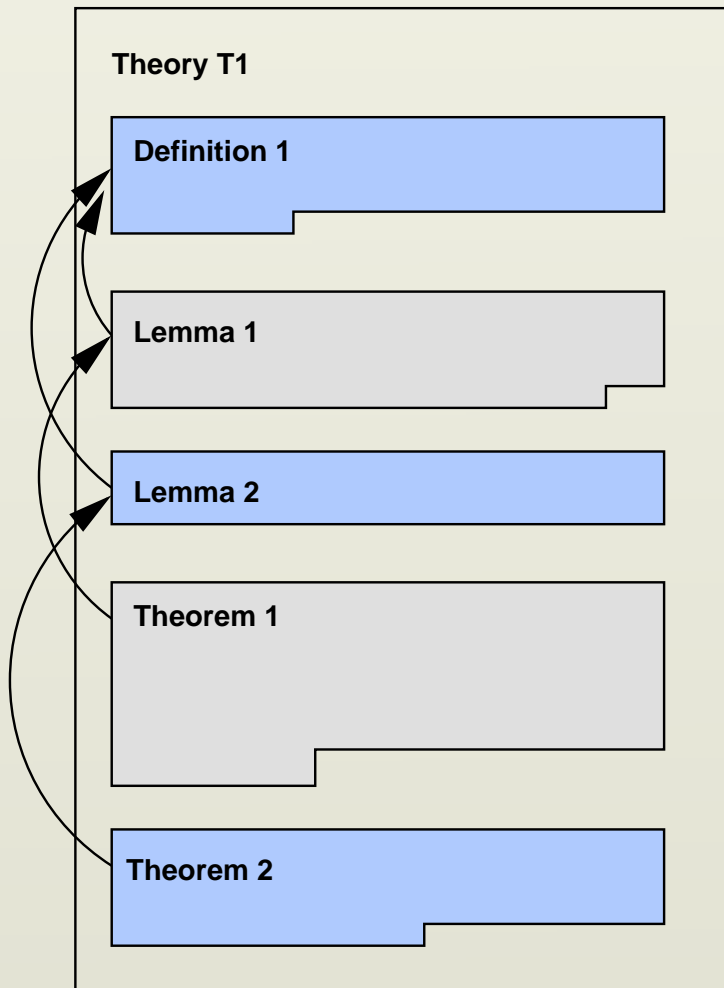
Linear dependency:

- Classical script management
- No prover assistance
- Potentially expensive

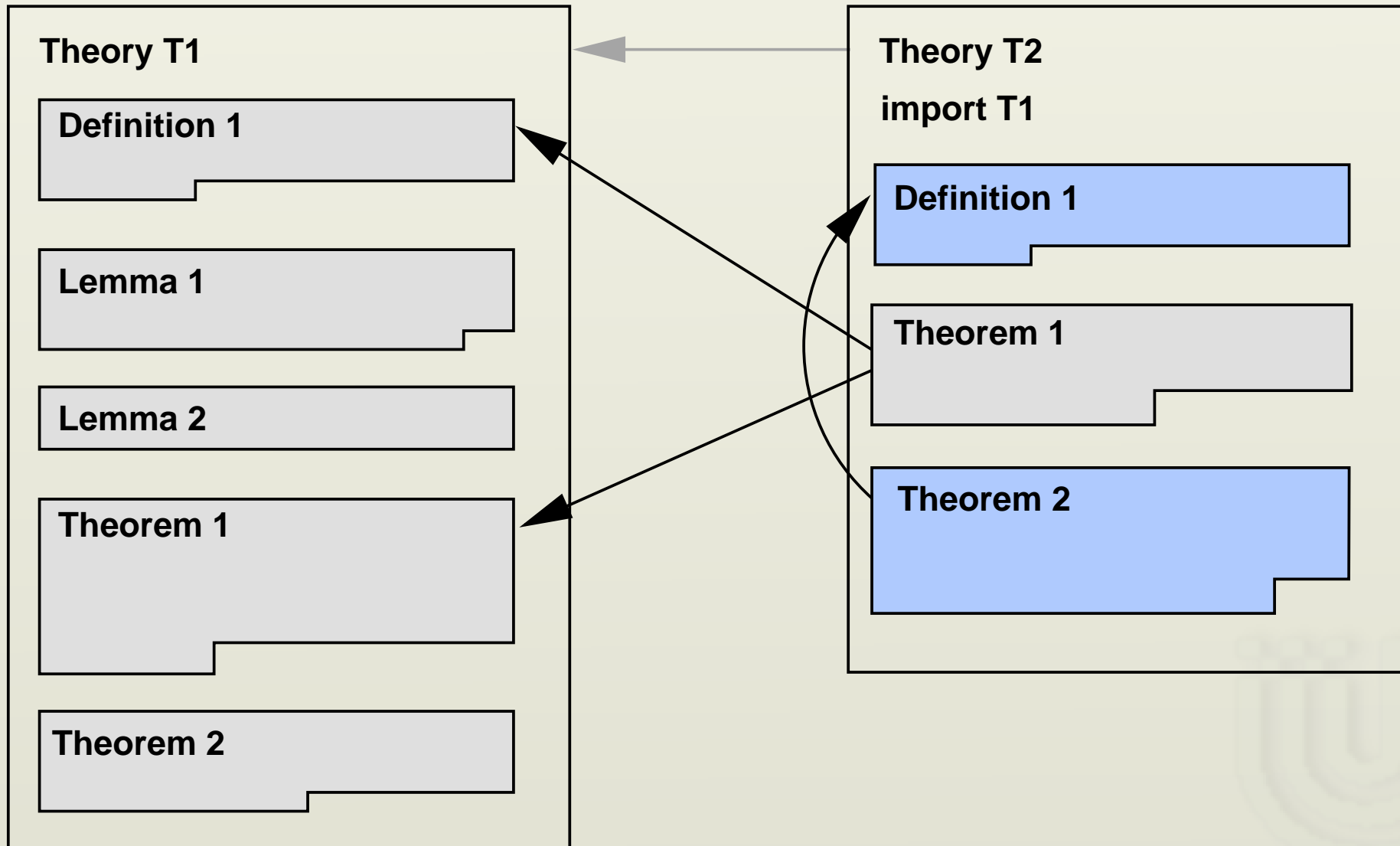
Theorem Dependencies

Explicit dependencies:

- Requires accurate prover assistance
- Better change management



Theory Dependencies



Final Remarks

- Proof scripts: native language plus markup
- Explicate structure — text split into commands



Final Remarks

- Proof scripts: native language plus markup
- Explicate structure — text split into commands
- Want abstract model of proof scripts
 - Institutions, parchments . . . ?
- Long-term goal: break linear view of proof scripts
 - ‘Proof graphs’?



Final Remarks

- Proof scripts: native language plus markup
- Explicate structure — text split into commands
- Want abstract model of proof scripts
 - Institutions, parchments . . . ?
- Long-term goal: break linear view of proof scripts
 - ‘Proof graphs’?
- State of Implementation
 - Broker, Emacs display: functional prototypes
 - Eclipse display: under development
 - Website: <http://proofgeneral.inf.ed.ac.uk/kit>