

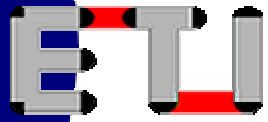
The Electronic Tool Integration Platform

Bernhard Steffen

University of Dortmund

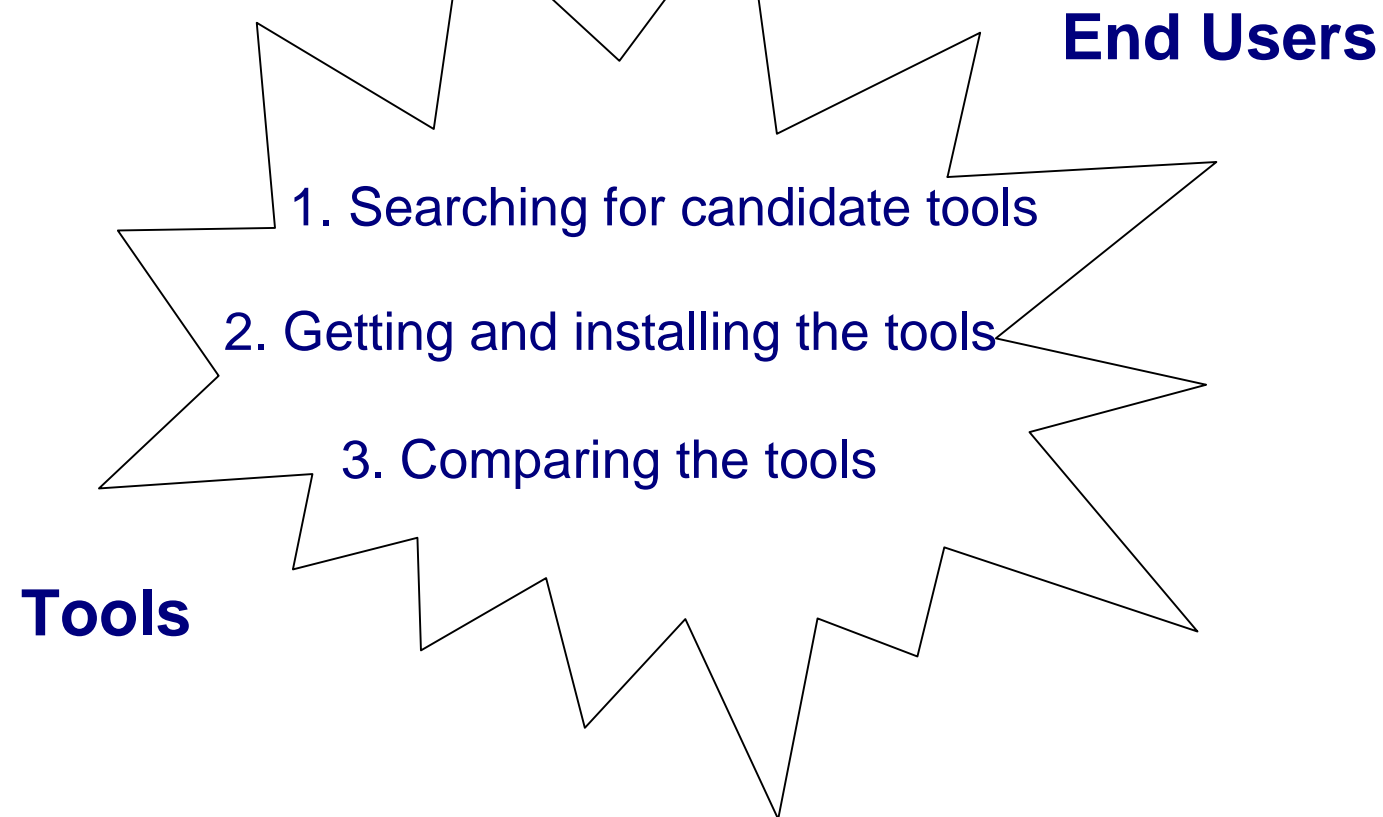
Volker Braun

Tiziana Margaria

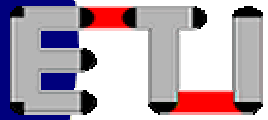


Roadmap

- Motivation
- Goals
- The **ToolZone** Software
- The **ETI Meta Model**
- Executing **Coordination Sequences**
- Conclusion



adequate communication channel is missing



Goals

The **ETI** Project is intended to support people who want to set up a **software tool experimentation site** by providing

- a Web-based, open platform for the interactive experimentation with and coordination of heterogeneous software tools



accessible by the **ToolZone** Software

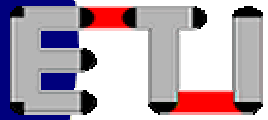
- an infrastructure organizing the platform development, extension and ETI site hosting



the **ETI Community Online Service**
at **www.eti-service.org**

The ETI Sites:

- application-domain-specific **instantiation** of the platform
- tool functionalities are located in the **tool repository**
- **tool providers** can **publish** their tools
- **end users** can **experiment online** with the tools and heterogeneous combinations of tool functionalities
- access to common case studies as well as private data space



The ToolZone Software

Tool Access

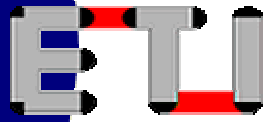
- **structured access** to the tool functionalities located in the tool repository with the ability to
 - get **detailed information** on each available tool feature
 - **execute single** tool features
 - **combine** heterogeneous tool functionalities **to programs**
 - **run the programs** via the Internet

➔ **tool coordination** and **Internet-based execution facilities** are the conceptual key features



Tool Coordination

- **full coordination** facility by means of the **HLL**
 - access to every available tool feature
 - mainly for experienced users
- automated **coordination support** by ETI's **synthesis component**
 - tool sequences are generated out of **goal-oriented** abstract
 - descriptions specifying **what** should be done instead of **how**
 - designed for unexperienced users
 - access to specifically structured functionalities



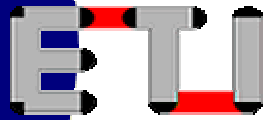
The ETI Meta Model

Activities

- ETI-specific **component model** models a tool feature as “**transformational**” entity

$$T_1 \longrightarrow T_2$$

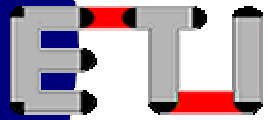
- **specified by two aspects:**
 - **interface aspect**
 - **functional aspect**
- can be executed in **stand-alone** or **tool-coordination** mode



The ETI Meta Model

Simple Text Processing Activities

| Activity Name | Input Type | Output Type | OS Cmd. |
|---------------|------------|-------------|---------|
| latex | TEXFile | DVIFile | latex |
| dvips | DVIFile | PSFile | dvips |
| gv | PSFile | Display | gv |

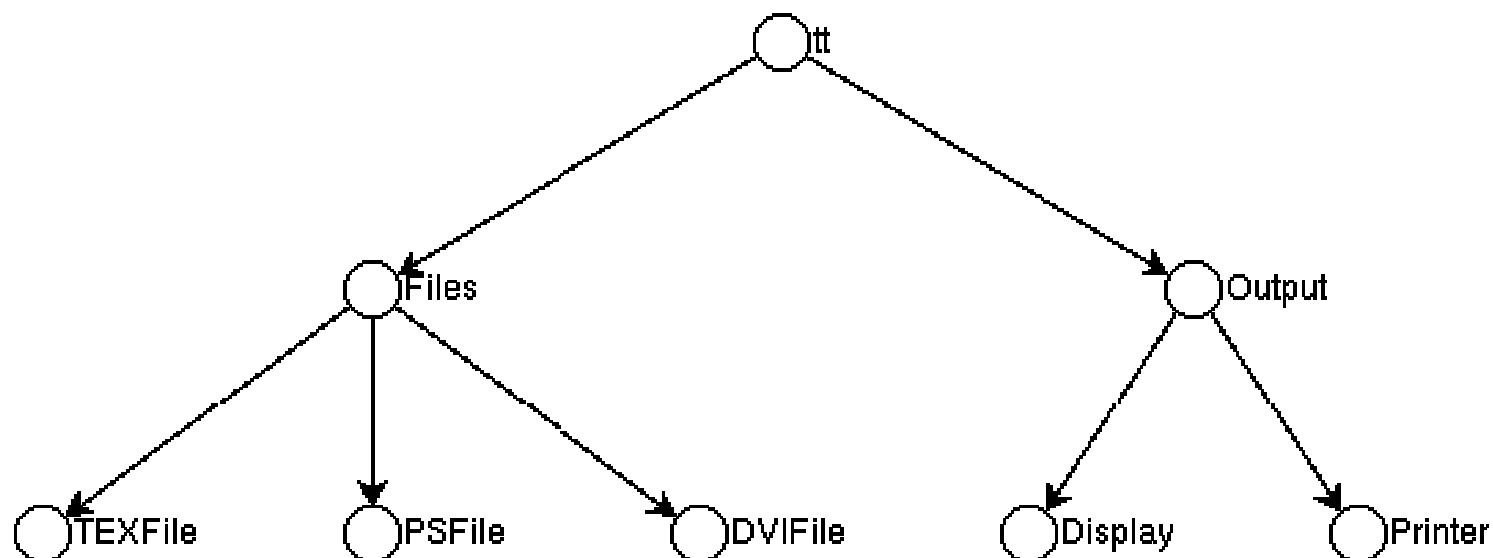


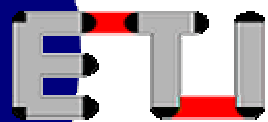
The ETI Meta Model

Taxonomies

- **classification** of types and activities
- represented as directed acyclic graphs
 - **leafs** represent **atomic entities**
 - **intermediate nodes** denote the **set of reachable entities**
 - edges **model "is-a" relation** between their source and target nodes

A Simple Type Taxonomy Example





GUI Impressions: Taxonomies

The Electronic Tool Integration Platform

File Edit Options Windows Help

```
registered file type ETI_FILE_MCL
** module 'MCLFile' imported...
registered file type ETI_FILE_XTL
** module 'XTLFile' imported...
registered file type ETI_FILE_SEQ
** module 'SEQFile' imported...
Loading /eti/platform/lib/METAFrame/hll/smv.hll
registered file type ETI_FILE_SMV
** module 'SMVFile' imported...
info: native methods of class ETIGraphHandler registered.
info: native methods of class ComponentGraphHandler registered.
info: native methods of class SynthesisGraphHandler registered.
info: native methods of class ActivityTaxonomyHandler registered.
info: native methods of class InterpreterImpl registered.
info: native methods of class SystemInspectorHandlerImpl registered.
info: native methods of class SystemGraphMenuHandlerImpl registered.
info: trying to connect to application server at //localhost:5000
info: getting connection handler with id 0 ... done
info: creating new interpreter ... done
```

The ETI Activity Taxonomy

File Edit Layout

```
graph TD
    model_checker((model_checker)) --> real_time_activities((real_time_activities))
    model_checker --> kronos_activities((kronos_activities))
    model_checker --> appal_activities((appal_activities))
    minimizer((minimizer)) --> misc_minimizer((misc_minimizer))
    minimizer --> strong_bisim_minimizer((strong_bisim_minimizer))
    minimizer --> weak_bisim_minimizer((weak_bisim_minimizer))
```

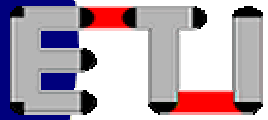
Activity Group tt

Description:

tt is the root of the ETI activity taxonomy graph. This group represents all group and atomic activities within the ETI activity taxonomy.

Please send suggestions and corrections to eti-team@eti-service.org
Last modified on 08-Mar-2001, 16:19

Execute Activity



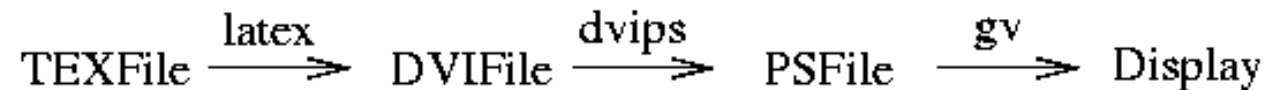
The ETI Meta Model

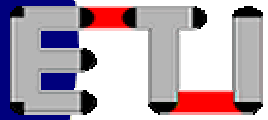
Coordination Sequences

- **sequential programs** built on the basis of the activities



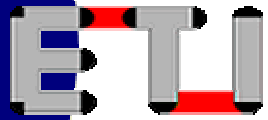
- can be **executed via the Internet** using the ToolZone software
- simple **example**:





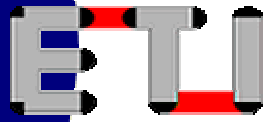
Loose Specifications

- building coordination sequences **manually** may be a **non-trivial task**
- **synthesis component generates** coordination sequences out of **loose** descriptions
- two orthogonal **dimensions of looseness**
 - **local** looseness
 - **global/temporal** looseness



Loose Specification: Example

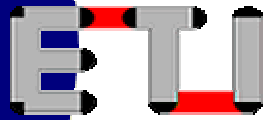
- “**TEXFile** < **Output**” to query all coordination sequences being able to display a **TEXFile** on an **Output** device
- **local looseness**: loose specification of the output device by using the type group *Output* instead of a concrete type, like *Printer* or *Display*
- **global looseness**: by using the *before* operator “<”



The ETI Meta Model

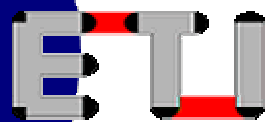
Generating Coordination Sequences

- ETI's synthesis component generates a **synthesis solution graph**
- based on a **coordination universe**
- **synthesis strategies** influence the resulting solution set
 - **all** solutions
 - all **minimal** solutions
 - all **shortest** solutions
 - **one shortest** solution

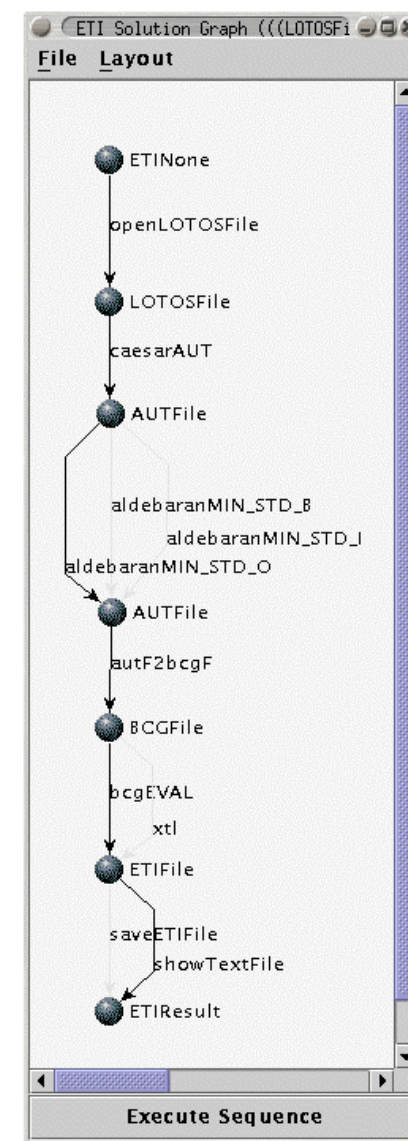
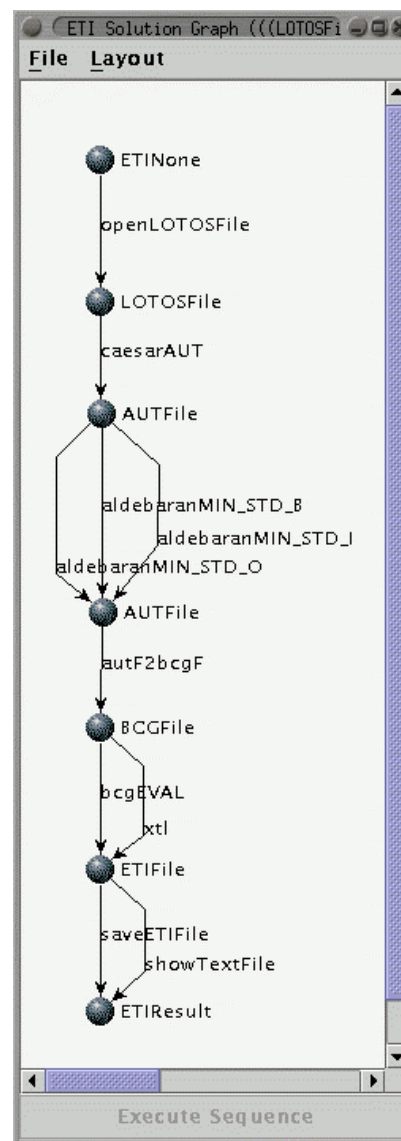
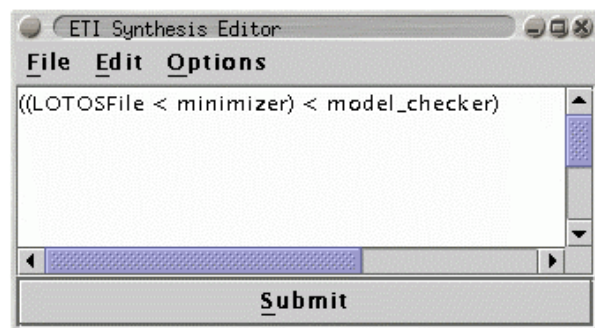


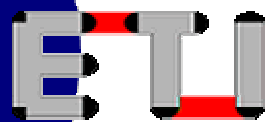
The Example

- The Alternating Bit
- Visual Verification
- Model Ckecking

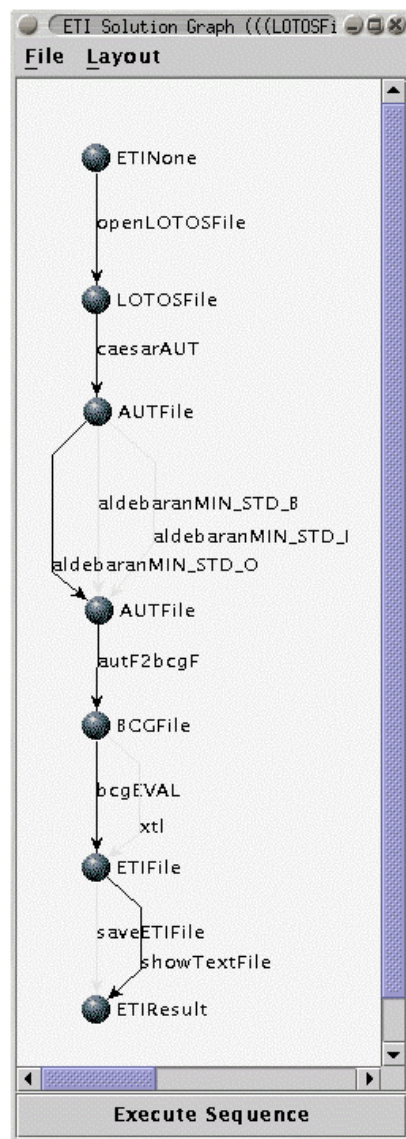


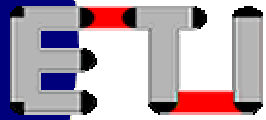
Synthesizing Coordination Graphs



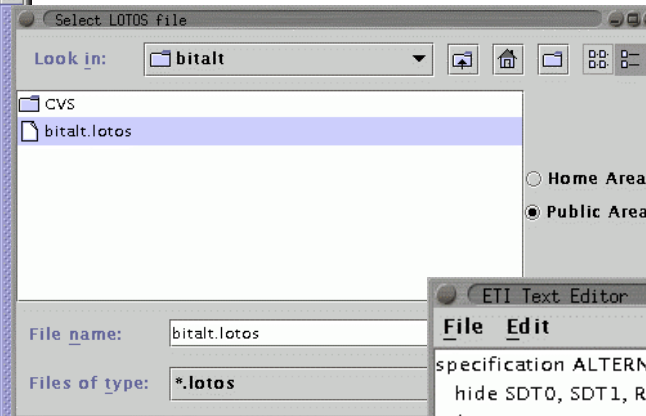
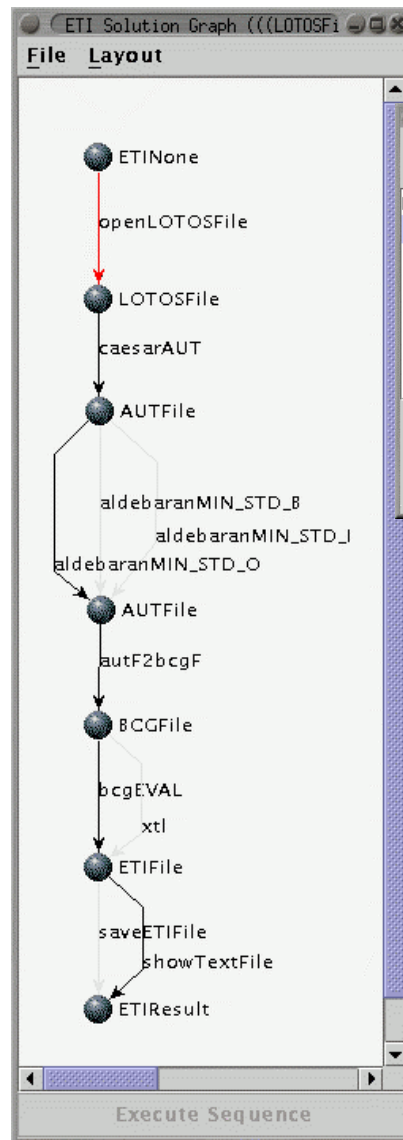


Executing Coordination Sequences





The LOTOS Specification



ETI Text Editor

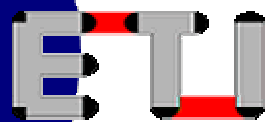
File Edit

```
specification ALTERNATING_BIT_PROTOCOL [PUT, GET] : noexit behaviour
hide SDT0, SDT1, RDT0, RDT1, RDTe, RACK0, RACK1, SACK0, SACK1, SACKe in
(
  (
    TRANSMITTER [PUT, SDT0, SDT1, SACK0, SACK1, SACKe]
    |||
    RECEIVER [GET, RDT0, RDT1, RDTe, RACK0, RACK1]
  )
  |[SDT0, SDT1, RDT0, RDT1, RDTe, RACK0, RACK1, SACK0, SACK1, SACKe]|
  (
    MEDIUM1 [SDT0, SDT1, RDT0, RDT1, RDTe]
    |||
    MEDIUM2 [RACK0, RACK1, SACK0, SACK1, SACKe]
  )
)

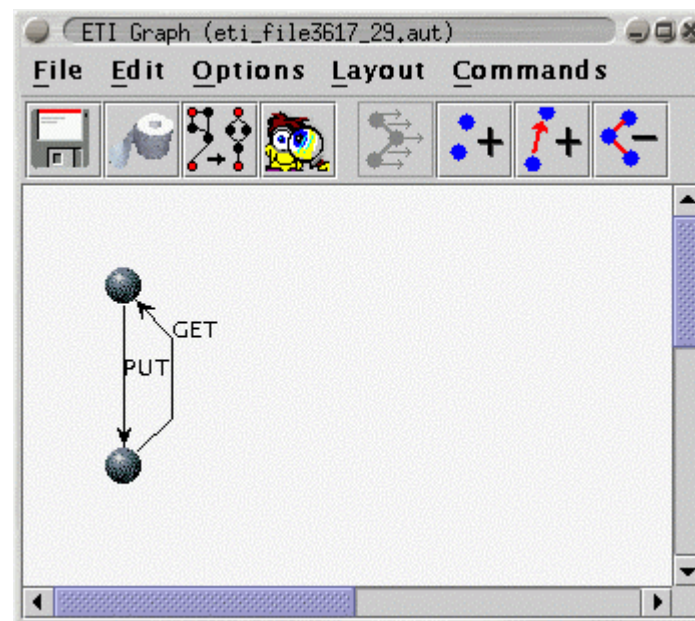
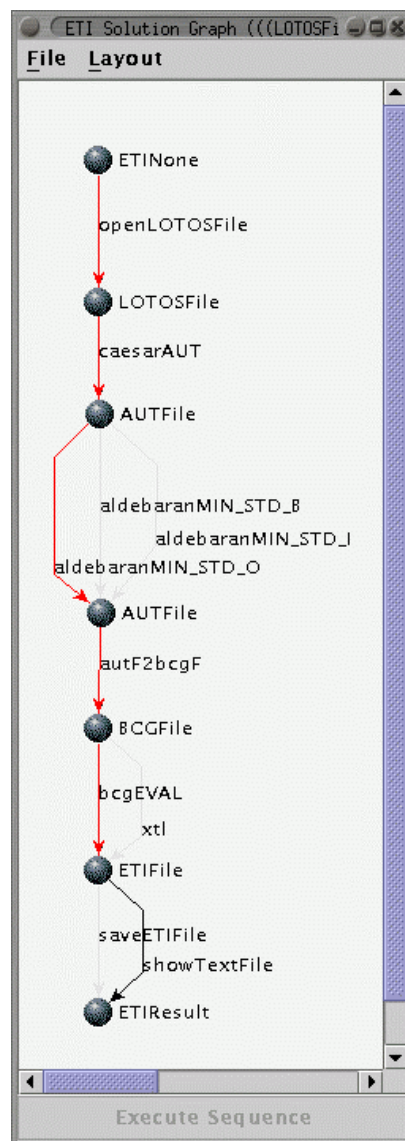
where

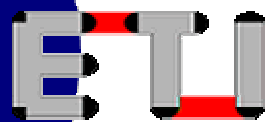
process MEDIUM1 [SDT0, SDT1, RDT0, RDT1, RDTe] : noexit :=
  SDT0;      (* reception d'un message *)
  (
    RDT0;    (* transmission correcte *)
```



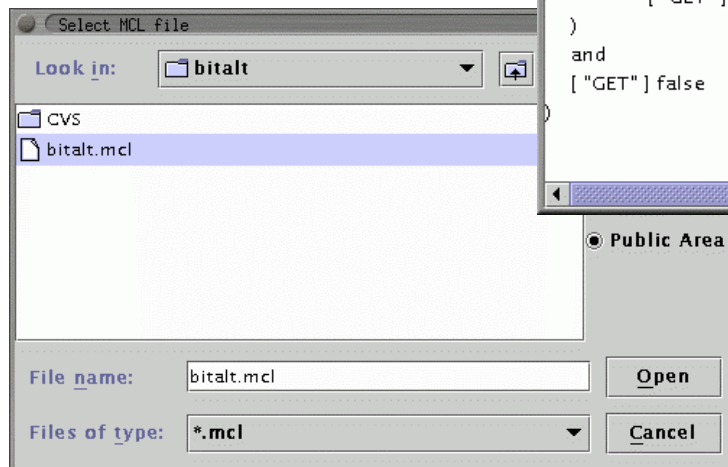
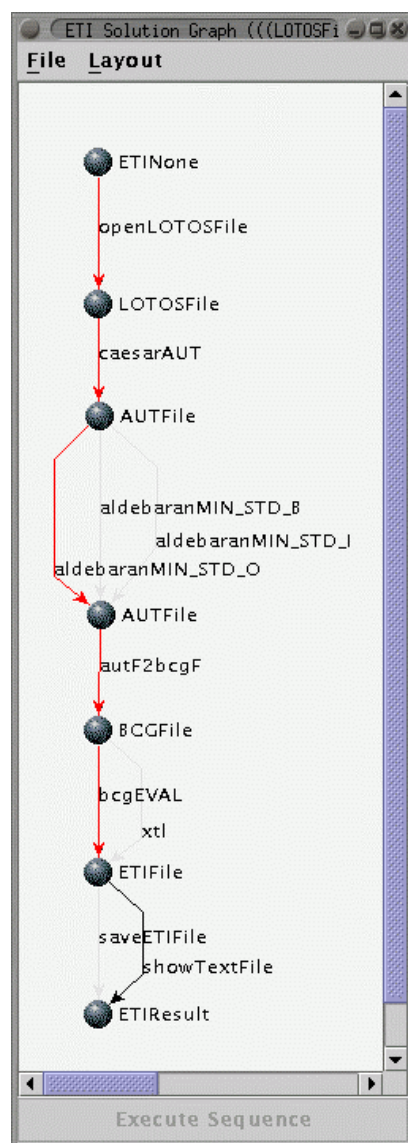



The Minimized Automaton





Program Execution

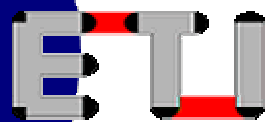


ETI Text Editor

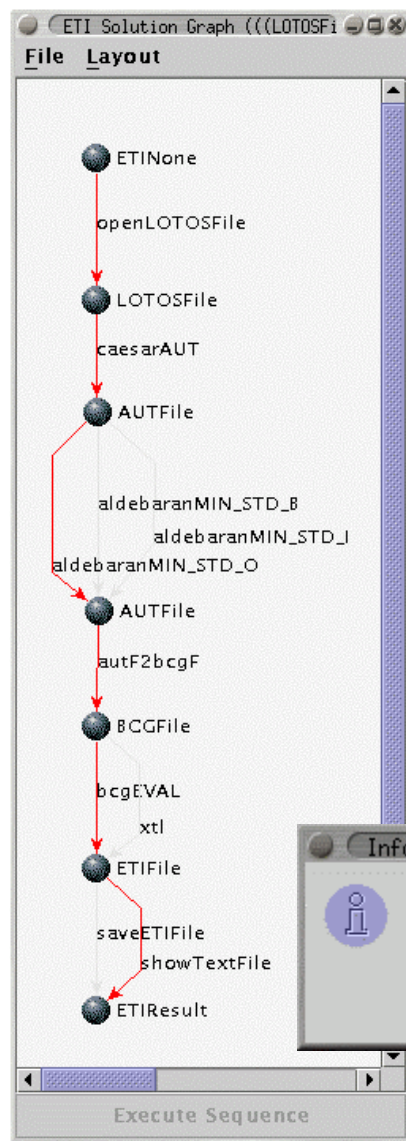
File Edit

```
(*
 * From the initial state, there is a strict alternation between
 * "PUT" and "GET" actions, starting with a "PUT". Moreover, after
 * every "PUT" (resp. "GET") a "GET" (resp. "PUT") is always potentially
 * reachable.
 *)

nu ExpectPUT . (
  < true* . "PUT" > true
  and
  [ "PUT" ] nu ExpectGET . (
    < true* . "GET" > true
    and
    [ "PUT" ] false
    and
    [ "GET" ] ExpectPUT
  )
  and
  [ "GET" ] false
)
```



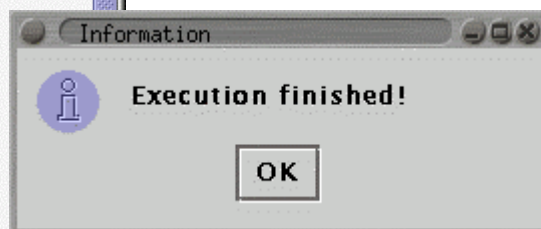
Model Checking

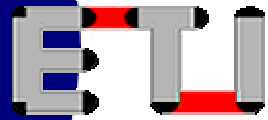


ETI Text Editor

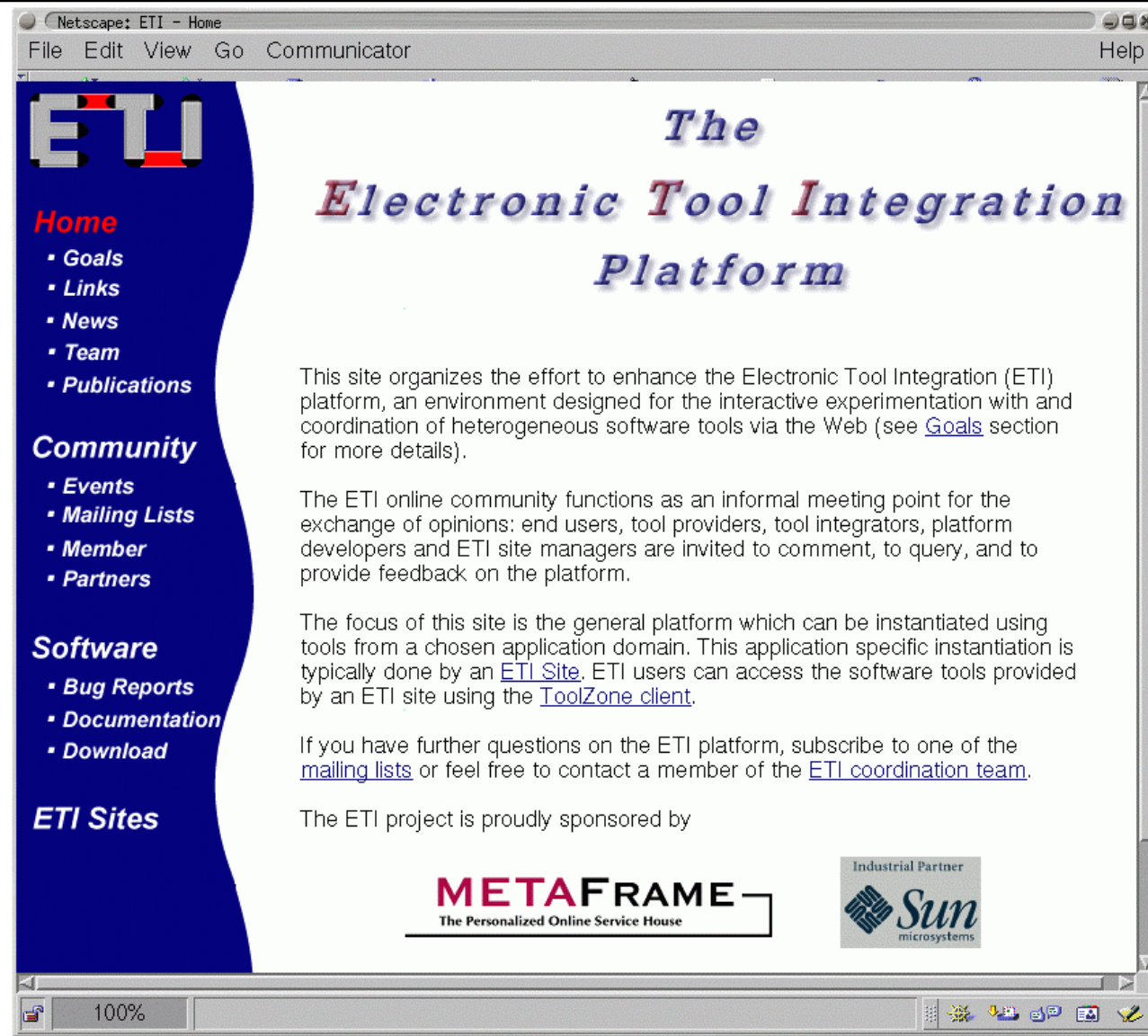
File Edit

```
bcg_open: using "/home/eti/tool-repository/tools/misc/cadp-99-x/bin.iX86/evaluator.a"  
bcg_open: running "evaluator bitalt.mcl" for "/tmp/eti_file3617_26.bcg"  
  
TRUE
```





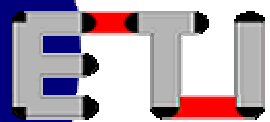
The ETI Community Service



- a **Web-based, open** platform
- interactive **experimentation** with heterogeneous tools
- elaborate **coordination support**
- coordination **programs** can be **run via the Net**

- **The ETI Community Service:** support for people, who want to set up a **Web-based communication channel** between **tool providers** and **end user** see: **www.eti-service.org**
- a concrete instance: **International Journal on Software Tools for Technology Transfer (STTT)**
- **We are in the course of setting up**
Network of Excellence

The Electronic Tool Integration Platform



Logical Layers

Client Layer

ToolZone Client



RMI/HTTP

Internet Access Layer

Internet Access Server



JNI

Feature Layer

Tool Management Application

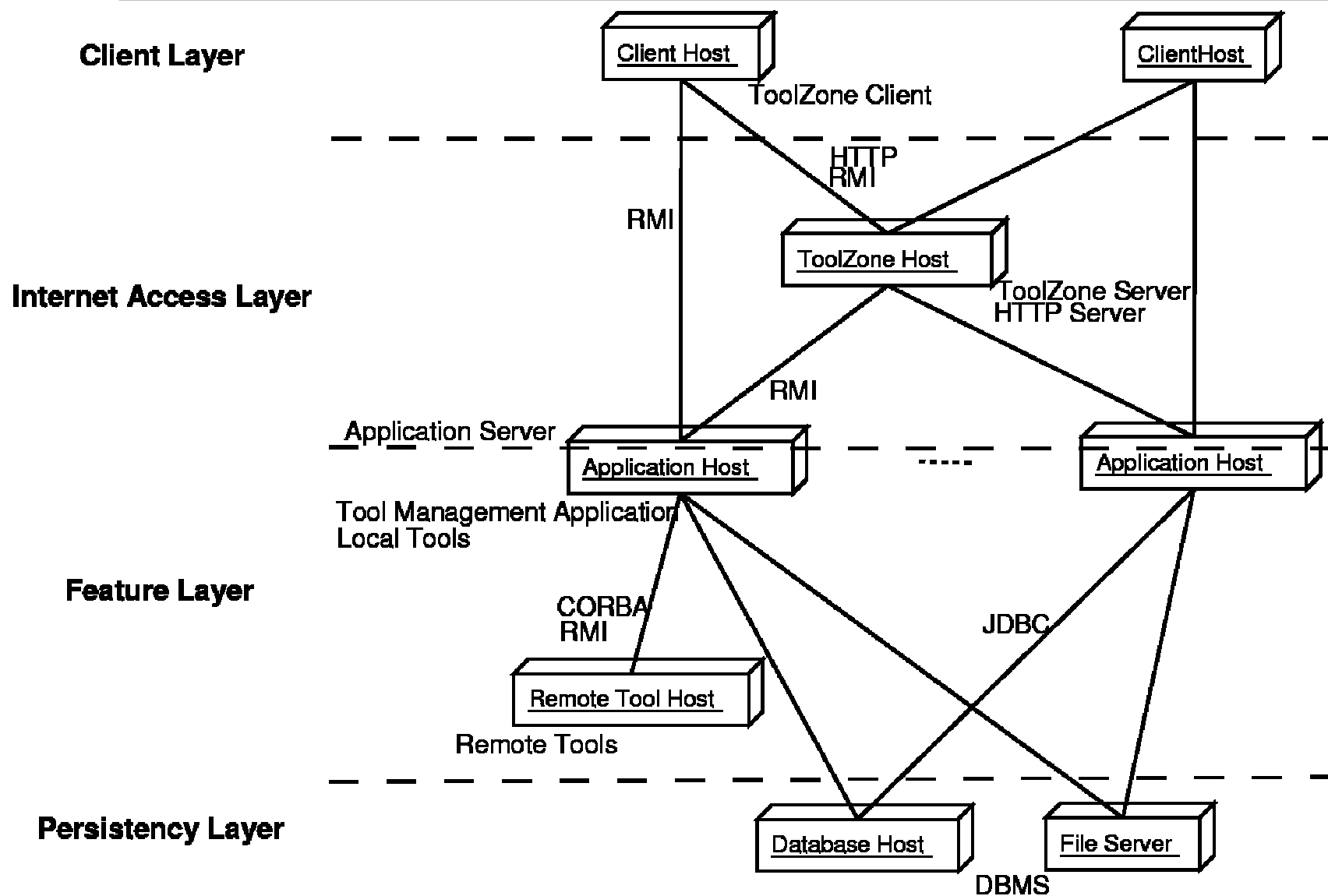


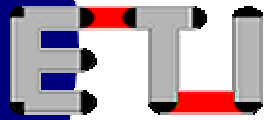
JDBC/FS/CORBA/RMI

Persistency Layer

Tool Repository

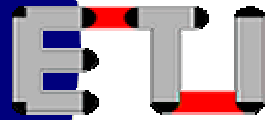
Software Architecture





The Tool Management Application

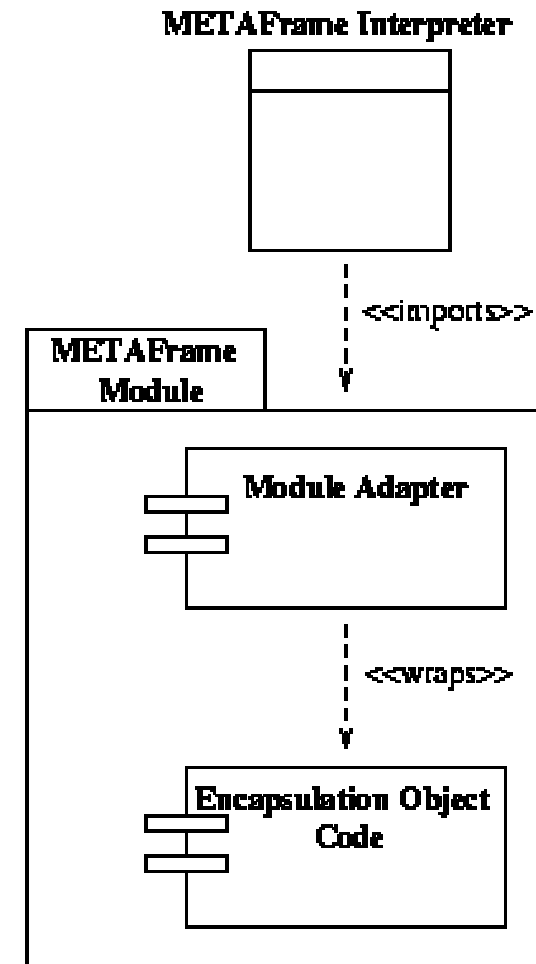
- **fully controls the access** to the tool functionalities located in the tool repository
- based on the **METAFrame environment**
- **application-independent kernel** managing the **application-specific components** (here tools features)
- the kernel provides:
 - the **hypertext system**
 - the **synthesis component**
 - the **HLL interpreter**
- central for **tool integration**



The Tool Management Application

The High-Level Language

- **procedural** programming language
- can **dynamically be extended** by new basic types and procedures
- types and procedures are provided by **METAFrame Modules**





The Tool Management Application

Tasks for Tool Integration

- **split up the tool** to be integrated into set of activities
- **classify** activities and types within ETI's taxonomies
- **implement METAFrame Modules** which HLL-enable the chosen tool functionalities and types
- **write HLL program fragments** specifying the operational view of the activities