

# Isabelle HOL Models for the Intentional Naming Service

CS576 Spring 2005 Project Summary  
Prof. Elsa Gunter  
Student: Marcelo d'Amorim

---

May 11, 2005

## 1 About INS

Intentional Naming Service (INS) is a protocol designed to facilitate the discovery of resources on a computer network. Instead of informing an IP address or the logical name of a resource, the user is asked to inform what he desires to get from the network via a *query*. In return, INS responds with the set of resources that satisfies the constraints the user informed. INS is essentially a *database* which associates *announcements* describing capabilities to *resources*.

In this work we are interested in proving the correctness of the current [1] INS implementation.

## 2 Approach

Instead of proving directly the correctness of functional properties, we aim at proving that a concrete model of INS is a valid refinement of an abstract model. Therefore, any property that might be of interest about INS can be proved on the abstract model provided that we prove the refinement relation.

The interface of INS contains three operations: **empty**, **lookup**, and **get**. The first creates an empty database. The second gives a set of resources when provided with a database and a query. The last operation, when given a database and a resource, will return the announcement stored in the database associated to the given resource. Note that this operation is partial.

We modeled the abstract INS database as a finite map comprised of tuples of the form  $(r, an)$  for resource  $r$  and announcement  $an$ .

The concrete (real) database [1] is represented as a directed acyclic graph (DAG) – for space and query-time efficiency – encoding very compactly the partial function “logically” associated to databases.

### 2.1 Organization

The INS theory is divided into the following files:

- `INSSharedDataTypes.thy` : Data types used in both models such as attribute, value, resource, announcement, and query.
- `SimpleINS.thy` : The abstract model of INS databases. It includes a simple proof.
- `RealINS_base.thy` : The concrete model of INS along with the theorem: "`getAllResources dba = dom (alpha dba)`", where `dba` stands for any concrete database, `getAllResources` is a primitive and mutually recursive definition, `alpha` is the abstraction function, and `dom` gives the domain of a function as a set.
- `INSValidityDataTypes.thy` : Includes (assumed) sufficient conditions for the validity of queries, databases, and announcements.
- `RealINS.thy` : The definition of `lookup` and the proof that the concrete database is a refinement of the abstract w.r.t. `lookup`.

### 3 Progress

We have modeled the abstract and concrete databases, proved one property on the abstract model; and simplified the problem of proving refinement of the lookup function by proving and creating some auxiliary lemmas.

### 4 Difficulties

We realized that Isabelle does not generate “clean” (not involving the measure function) induction principles for mutually, and non-primitive recursive definitions. We had to come up with one manually. This delayed the proofs to some extent.

Almost all definitions we have are mutually recursive. It means in most of the cases structural induction proofs can not be given in isolation. That is, if we want to prove that  $P$  holds for the database, we need to come up with similar facts for every substructure of the database and prove them altogether. In practice, this leads to an unexpected burden on proofs, especially of lemmas.

### References

- [1] S. Khurshid and D. Jackson. Correcting a Naming Architecture using Lightweight Constraint Analysis, 2001. <http://sdg.lcs.mit.edu/pubs/2001/correctingNA.pdf>.