

## (define insecure-mode #f)

pronounced: *Define insecure mode false.*

A line you **want** to have in your configuration file.

### Knowledge brings Responsibility - or - The Proof Of Innocence

We all know: If someones actions are supposed to incur binding consequences, i.e., responsibility, the person must know (or at least be able to know) what those actions will cause. This implies one can proof them that there was never a surprise hidden in any device the person used in the action. Let's look at an example to illustrate the point: If one browses through [eBay](#) just to find what's available and suddenly finds out that the web browser the person used sent "buy right now" requests for all the goods... who's responsible? -- Certainly not the person, since the tool had a secret.

Since we need that proof to continue our daily business, we must be careful to avoid a situation, where it becomes impossible to adduce that proof. However in 2004 we might soon reach that point at least with respect to "e-bussiness":

In my [contribution](#) to the [Software Patent Legislation Benchmarking Conference](#), European Parliament 14th May 2004, I referred to the configuration variable `insecure-mode` of the [BALL](#) source code. To my understanding, the meaning of this variable happens to match the importance of the article 3 of the amendments, which the european parliament made to the original software patent directive proposal. As a consequence, I urged to keep the article 3, which endorses the understanding that "data processing" is **not** a "field of technology".

### How can a variable have such a meaning?

If data processing (as such) was a field of technology, the notion of private property on procedures (as such) became conceivable. But such private procedures are useless:

[BALL](#), a first shot on an implementation of the Askemos concept, happens to model basic legal concepts, and must do so, since it shall eventually work within legally binding contexts. One such basic concept is the "proof of innocence" for any [facility](#) within the network: For all facilities, which are supposed to work in the name of a user, the system must produce an undeniable proof to this user beforehand, that the facility will follow a certain procedure (especially, that they will not alienate the users rights against the users intentions). The user is free to choose the appropriate treatment (from disabling some rights up to refraining from access at all) for facilities that fail such proof.

Within Askemos the [action document](#) defines the procedure in question. Since a wide variety of procedures are obviously useful (ranging from "simple", arbitrarily mutable files, weblog style controlled access up to sealed, immutable information), that document can be chosen, but certain restrictions apply. These restrictions match the restrictions required for legal code: they must be either available under pre-approved conditions or guaranteed to be immutable while they are applicable. With the help of that recursive definition we can rely on the analysed meaning of the code and therefore predict consequences of actions. Once we have a first guaranteed tamper proof procedure, we can grow a pool of approved procedures, since one can trust that the results of prior analysis still apply.

## DefineInsecureModeFalse

The catch is in during system initialisation, when a first procedure must be approved (it carries the usage license, in a way the equivalent of a constitution). Because there is no other document yet, it must reproduce and approve itself. This is, where the configuration variable `insecure-mode` comes into play: the relevant part of the code is around [line 234-240<sup>\[1\]</sup>](#), where - depending on the value `insecure-mode`, the default code is either set to be "private" (`my-oid`) or "public" (`public-oid`). The only difference between the two procedures is, that the former code will allow modifications at any time (which is equivalent to the notion of "private property on the piece of information"), while the latter makes it an immutable, tamper proof deed.

The consequence: the first way of initialisation allows some more flexibility with respect to complete reconfiguration and rewrite of all rules. The easy of modification can be handy from a technical and development point of view, but it renders the whole system useless for legally binding contexts, since any operation can be denied later and therefore no user may ever be held responsible for events, which where apparently caused by that user.

### Relationship with Article 3

Article 3 requires to understand that data processing is **not** a "field of technology" (since if it was, patent monopoly where applicable). If procedures (algorithm, procotols etc.) where under arbitrary control of individuals ("private", i.e., patented), they where equally useless for legally binding contexts. Consequentially data processing could never become legally applicable. e-Business, e-Government, e-Health - all those ideas became a cynical joke.

---

1

Line numbers refer to the first cvs check in at sourceforge.net in October 2001 several month past the first public release.

---

Last modifikation: Sat, 24 Apr 2004 12:21:36 +0200

Author(s): jfw,

Document number A849640f672ed0df0958abc0712110f3c page DefineInsecureModeFalse delivered to public at Wed, 08 Sep 2010 16:28:21 +0200