Draft XMILE extension to capture AI usage information

AI Information

The <ai_information> tag is a root level tag used to describe status and settings related to the use of AI in model generation. Its purpose is to ensure that someone opening a model can determine reliably whether, and where AI was used in model creation. While this will not prevent someone from developing a model using AI then transcribing the information to another model, it does mean that use of AI in the software, including through copy and paste, will be tracked.

The <ai_information> tag has three subtags:

<log> is the log of interaction between the user and the AI agent. This should be simple text, markup, or markdown in whatever form is most convenient for the software. There will be a flag under the <status> tag indicating whether this log has been truncated. This tag can be either empty or missing if there is no log of activity.

<settings> are software specific settings related to AI usage. No subtags or attributes are specified here as the nature of those setting is expected to be software specific.

<status> is used to indicate whether AI has been used and where. This information is persistent and becomes invalid unless it is updated each time a model is written. This information is also used when working with copy/paste or other injection of model content as noted below. It consists of attributes, but no subtags.

The following attributes are supported under <status>. Additional attributes may be included as long as they follow the rules for signature generation as described below.

Tag	Valid values	Comments
verified	true/false	If false, the content is not trusted. In this case the other attributes may be skipped, as it indicates the use of AI in model construction is unknown. Models saved in software not supporting <ai_information> will have verified false.
trusted	true/false	True if all required tags have been specified and the signature on last model reading was correct as described below. Typically false will arise from bad signing or a public signing key that is not trusted.

tampered	true/false	True if the signature verification has ever failed when reading the model. When false trusted will also be false.
uses	true/false	True if AI has ever been used as part of model construction.
analyzed	true/false	True if any AI based analysis of the model has been done, and the model subsequently saved.
log_truncated	true/false	True if the log of all interactions with the AI agent has ever been truncated. If the software does not store this log as part of its regular activity this should be set to true.
timestamp	integer	The time at which the ai_information was last saved (seconds since Jan 1, 1970).
algorithm	ed25519	The algorithm used in the generation of the signature. Only one algorithm is supported.
keyurl	https URL	A web address that will return the base 64 encoded public key for verifying the signature in response to a get.
want_var_info	true/false	This is true when variable information is included in the text used to generate the signature.
want_var_status	true/false	This is true when the status of each variable in the model is included in the text used to generate the signature.
skip_module_files	true/false	If true, then modules saved as separate files will not have their variable info included but instead will include it in their own ai_information section. If false, or missing, all variables in all modules will be processed when want_var_info is true.
sampling	Integer	The number of entries between variable information. This is used for large models (recommended > 500 variables) when generating all variable information might be time consuming. If left off it should be assumed to be 1.
want_log	true/false	This is true if the AI log should be included in the text used to generate the signature. If AI has not been used this would normally be false, even though true would generate the same signature.
signature	String	The base 64 encoded signature for the text.

Signing

Signing is done using the ed25519 signature algorithm against a message that is constructed by combining the attributes above along with the additional information

indicated. The text to be assigned should have all whitespace (spaces, tabs, line feed, carriage return) as well as underbar (_) characters removed. It consists of

1. All key/value pairs from the <ai_information> tag sorted alphabetically by key. There should be an = between the key and the value and space after the value (but no spaces within either the key or the value). The attribute values should not be quoted – so for example log_truncated="true" in the xml becomes logtruncated=true).
2. If want_var_info is true, then then the variable name followed by = then its equation for the model variables sorted alphabetically by variable name. For models containing modules, variables in the root module should appear first followed by the module variables with the variable names module qualified. Modules and sectors (and any other content other than stocks, flows, and auxiliaries) should be excluded from the list. For arrays with non apply to all equations all equations should be listed from the first to the last element with no separation between. This content may be partial based on the sample tag's value. Each equation (or equation list for non apply to all equations) should appear as it does in the XML, though there are no spaces within either the equation/equation list or the variable name. Equations using cross level ghosts should use the local variable name. The equations for cross level inputs should appear as they do in the xml even when such equations are empty or invalid.
3. If want_var_status is true then the AI status information about each variable (this ignores sampling) when the status is a single letter from the table below. The full list of status values should be followed by a space, but there is no space in the list.
4. If want_log is true then the content to be signed will have the log appended, again removing all whitespace characters. The log itself must be included in the <log> tag.
5. If there is a terminal space on the constructed message string (which will happen if the log is not included) it should be removed. Thus, the only spaces are internal separators and are never from the content forming the message.

Codes for different AI usage by model variable

A	No information. The variable is from an untrusted source. Trusted must be false if any variable has this status.
B	Created by a person not using AI. Will only occur when a modeler adds content using AI to an existing model. (Depending on the software implementation these may always be reported as F).
C	AI was used to generate the variable and its equation. No editing by a person has been done.

D	Created by a person but edited by AI. This may always be H depending on the software implementation.
E	Edited by a person with unknown creation. This should never occur.
F	Created and edited by a person not using AI.
G	Created by AI then edited by a person (though possibly edited again by AI).
H	Created and edited by a person and also edited by AI.

Note that the above states do not incorporate any sequence information. For example, created and edited by a person then edited by AI then edited again by a person is the same as if that last editing had not occurred (H).

Each of the above flags should also appear in the variable definition (inside of <model><variables > using the <ai_state> attribute > as in:

<flow name="ai generated flow" ai_state="C">

Signing

Signing should be done using the ed25519 signing algorithm (https://datatracker.ietf.org/doc/html/rfc8032) with the private key corresponding to the public key returned by key_url. Software vendors may choose which public keys they trust. An untrusted key shall be treated as an unsigned model, though checking for tampering may occur.

Notes on settings

The AI settings are all irreversible. If a model starts unverified, or has content injected through the clipboard or other mechanism that is unverified, the model will be forever unverified. Same for (not)trusted, uses, analyzed, log_truncated.

Software implementations should require want_var_info, and additionally want_var_status if ai has been used. Unless the log has been truncated want_log should also be true.

When injecting content from outside sources such as the clipboard, software implementations should include the signing, but may exclude variable information as long as the source of the content is fully trusted (most typically the software itself).

Software providers should communicate amongst themselves to determine bilaterial standards for key trust.

Examples:

Status information for a simple AI generated model with 1 variable price:

<status verified="true" trusted="true" tampered="false" uses="true" analyzed="false" log_truncated="false" timestamp="1751663454" algorithm="ed25519" keyurl="https://iseesystems.com/keys/stella01.txt" want_var_info="true" want_var_status="true" want_log="true" sampling="1" signature="FB3fCMap7xCQ+9GUmxl1QvdAQDehvMyPSOMNYtjv5hCWh6cHSA249hn6fo9120YAhmEUIoVg7hqMcs0aLaN/BA==" />

The message string used to generate the signature is (line wrapping occurs at the embedded spaces, but there is no space at the end):

algorithm=ed25519 analyzed=false keyurl=https://iseesystems.com/keys/stella01.txt logtruncated=false sampling=1 tampered=false timestamp=1751663454 trusted=true uses=true verified=true wantlog=true wantvarinfo=true wantvarstatus=true price=50 C <h3>Youinstructedmeto...</h3>
<h3>HereiswhatIhavedone...</h3><h3>SimplePriceModelNoFeedback</h3>Themodelwasupdatedtoincludeasinglevariable,'price',asrequested.Nofeedbackloopscouldbeformedbecausetheinputonlyspecifiedasingle,constantvariable,andnoothervariablesorrelationshipswereimpliedorprovidedasevidencetosupportthecreationofadditionalvariablesortheircausalconnections.Therefore,nochangesweremadetoclosefeedbackloops.

And the variable definitions for price is:

<aux name="price" ai_state="C">
<doc>The price of the product, set at a constant value as per the request. There is no information provided to suggest that this variable should change over time or be influenced by other factors.</doc>
<eqn>50</eqn>
<units>$/wheel</units>
</aux>

The C in ai_state matches the C just after the equation for price (50) in the signature string.

The same content for a model that was built entirely by hand would look like:

<status verified="true" trusted="true" tampered="false" uses="false" analyzed="false" log_truncated="false" timestamp="1751663148" algorithm="ed25519" keyurl="https://iseesystems.com/keys/stella01.txt" want_var_info="true" want_var_status="false" want_log="false" sampling="1"

signature="hnBrID6G4q9cX3Ga2WPHmFIMpVGCrTlLcxpFKU2stKNuL7/8IhOvZVDxv
ovCJ77AlRcfN4abQrWv9JBQ7t4WBQ==" />

Note that want_var_status is false, as the variables are all untouched by AI. In this case the content string to be signed contains only the key-value pairs and the list of variables and equations.

algorithm=ed25519 analyzed=false
keyurl=https://iseesystems.com/keys/stella01.txt logtruncated=false sampling=1
tampered=false timestamp=1751663148 trusted=true uses=false verified=true
wantlog=false wantvarinfo=true wantvarstatus=false price=50

The definition for price is largely the same, though no documentation was added and it is unmarked because the model records no AI usage.

```
<aux name="price">
  <eqn>50</eqn>
  <units>$/Wheel</units>
</aux>
```