Vendor Information

LitSavant Ltd	
Mark Dingle, 020 8923 4333 Support@litsavant.com	
70 Leyton Park Rd, London E10 5RL	
Company Website.	

Company Description

A UK technology firm specialising in eDiscovery and litigation support services across a range of tools and platforms.

Founded by Mark Dingle in 2010 to assist with the practical application of technology to eDiscovery and litigation support, we first specialised in the use of Relativity™. As part of our consulting practice, we realised that the Relativity platform was missing a simple method of building logical rules to check coding as it happened, rather than searching for inconsistent coding results after the event.

In response to this need, we developed the LitSavant Conformity Engine – a simple, but powerful Relativity application which performs checks on data entry as it happens, to ensure that the information entered is consistent with logical rules.

We currently focus on the distribution and support of this application, as well as providing consultancy services on eDiscovery and litigation support projects.

Software Information

LitSavant Conformity Engine

The LitSavant Conformity Engine is an application that works within Relativity, which enables Project Managers to design and implement validation rules and other logical processes which are then enforced during data entry.

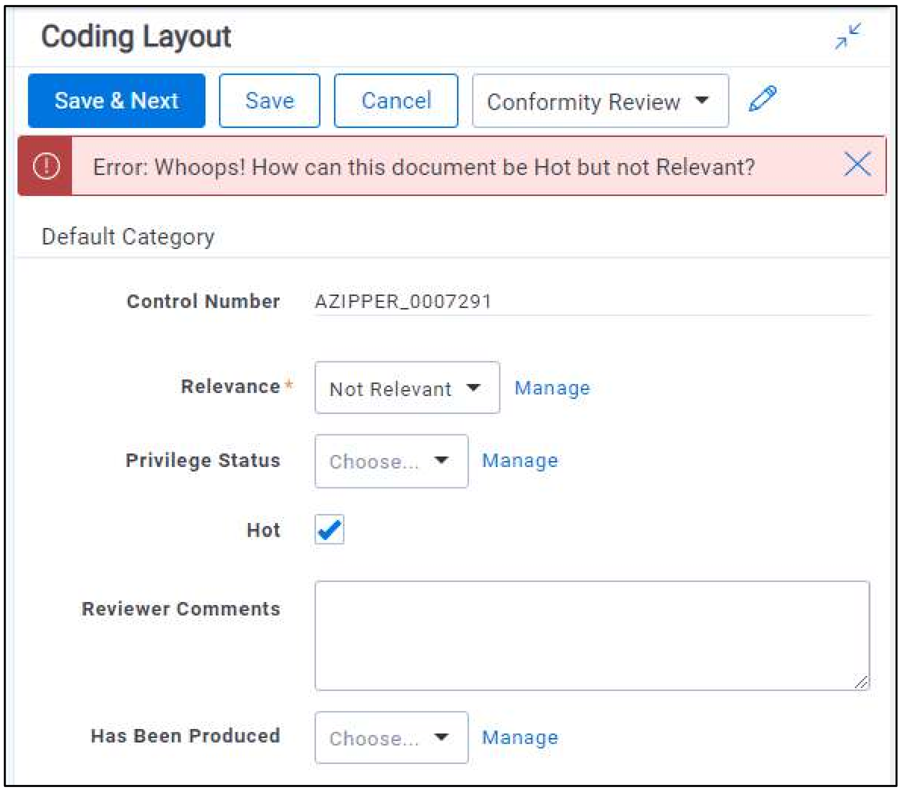
This works by allowing the user to specify what they want the software to do and the conditions that need to be fulfilled, in order for the software to do it.

Here are 5 easy examples illustrating how these rules can be used and why they are very useful in increasing accuracy and reducing costs:

(Examples on Following Pages to Allow for Graphical Depictions)

Example 1 – Alert the user when they get it wrong

This is the commonest scenario and results in an onscreen message when the user codes something in a way that doesn't meet predefined rules.



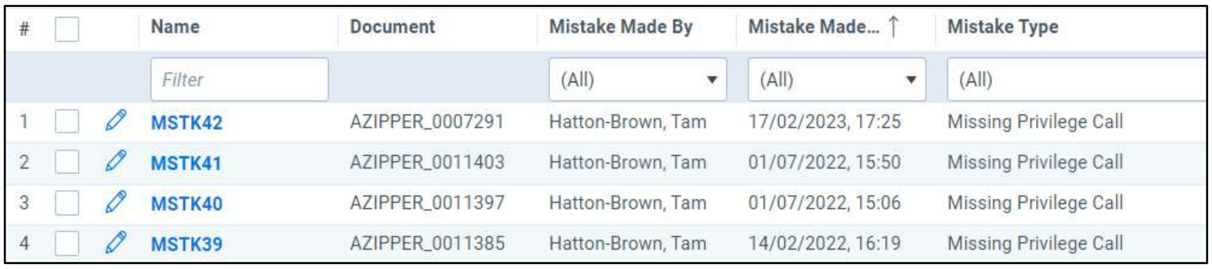
Error: Whoops! How can this document be Hot but not Relevant?

Default Category

Control Number	AZIPPER_0007291
Relevance*	Not Relevant
Privilege Status	Choose...
Hot	✓
Reviewer Comments	
Has Been Produced	Choose...

Example 2 – Log mistakes (when users get it wrong)

When a user makes a mistake (such as coding a document both "Hot" and "Not Relevant" as per the example above), in addition to alerting them so that they correct the error, we can also log the mistake. The mistakes can then be analysed to identify recurring themes for corrective action etc.



#	Name	Document	Mistake Made By	Mistake Made...	Mistake Type
	Filter		(All)	(All)	(All)
1	MSTK42	AZIPPER_0007291	Hatton-Brown, Tam	17/02/2023, 17:25	Missing Privilege Call
2	MSTK41	AZIPPER_0011403	Hatton-Brown, Tam	01/07/2022, 15:50	Missing Privilege Call
3	MSTK40	AZIPPER_0011397	Hatton-Brown, Tam	01/07/2022, 15:06	Missing Privilege Call
4	MSTK39	AZIPPER_0011385	Hatton-Brown, Tam	14/02/2022, 16:19	Missing Privilege Call

Example 3 – Send an email when hot documents are found

When a user identifies and codes a document as hot for the first time, the software can send a customised notification email to specified users. The email can be customized to include information about the user who tagged the document as hot, any comments that they made, any additional coding from the document and a link to the document itself.

Example 4 – Update the "Last Coded By" field

It's not uncommon to want to know who reviewed a document during a particular review round. The software can update a "Coded By" field – saving the user from having to enter this information.

Example 5 – Update a “Family Privilege” field based on “Privilege” coding

Relativity propagation is a pretty blunt instrument in that the last coding decision on a family member is applied to all members of the family. If we want to know whether a family contains a privileged (or part privileged) document, standard Relativity propagation doesn’t really do the job.

Instead, we can create a custom action to use “Intelligent” propagation. With this approach, users code the “Privilege Status” of each document and the software then updates a field for all members of each family based on that coding to indicate if they are members of a Privileged, Part Privileged or Not Privileged family.

Detail

The examples above illustrate the 5 functions that the software can perform as part of the coding process. These functions are:

- Show an onscreen message
- Create a new instance of an object
- Send an email
- Update a field
- Execute a Relativity script

One or more of these functions can be triggered when the specified conditions are met. Each of these functions is customisable and can be triggered by one or a combination of conditions. The conditions themselves are also fully customisable.

Whilst all the examples above relate to coding documents, these functions can also be used on all custom objects.

Each of the functions above could be created by a programmer in code and deployed via an Event Handler. The innovation in the LitSavant Conformity Engine is that Relativity’s standard interface is used to enter the rules – no programming knowledge is required and so the rules can be created, tested and deployed in minutes. And because the LitSavant Conformity Engine saves your rules into your Relativity database, when it is time to upgrade Relativity, you simply update the application as part of the upgrade process. If you were using Event Handlers you would need your programmers to manually rewrite and redeploy each of your Event Handlers in all of your databases.

The LitSavant Conformity Engine is compatible with any version of Relativity Server supported by Relativity and with RelativityOne. It is licenced on an annual basis and is available to anyone with their own Relativity environment.