

# Unravelling Rules and Deciphering Dependencies

---

## Overview

Using a simple rule with no dependencies can be a breeze. It is easy to understand and to visualise. For example if your criteria is that the first name should start with an upper case letter this is both easy to implement and easy to follow. It is made even easier when you follow the logic using the built-in flowchart.

However not all rules are that simple. This document aims to give some understanding as to how rules work in combination with one another.

## Definitions

First here are some definitions of terms that we will be using throughout.

**Rule** – This is the criteria, the dependencies and the actions combined.

**Criteria** – This is the logic that acts on a specific field. This could be text based, date based, numeric or yes/no. It corresponds to the “Criteria” tab of the Rule.

**Dependency** – These are other rules that our current rule depends on. We can only show an error message if the criteria applies as well as the criteria on any dependencies.

## Unravelling Criteria

There are four different types of criterion depending on the type of field.

**Text** – e.g. first name, phone, gift type

**Date** – e.g. action date, attribute date, deceased date

**Numeric** – e.g. gift amount, total number of constituent codes

**Yes/No** – e.g. Has Valid Address, Is Primary Employee

Hopefully these should be relatively straightforward to use. Text based criteria make use of regular expressions. However, you should be able to avoid ever having to write your own as there are a number of different options available. When you *do* use the built in options, the regular expression is displayed for your reference and those who feel comfortable using them can edit the regular expression.

There are a number of other options that affect how the criteria work.

### How the criteria are applied

- Apply when the record agrees with the criterion
- Apply when the record disagrees with the criterion

This could be rewritten to say “Show the error message when the record agrees with the criterion” or “Show the error message when the record disagrees with the criterion”.



---

This definition is fine for simple rules. However it does not mean much when this rule is a dependency and not going to show a message. In that case we could rewrite to say:

“Allow the parent rule to show the error message when the record agrees with the criterion” or “Allow the parent rule to show the error message when the record disagrees with the criterion”.

More about dependencies later and this will be clearer.

### Applying to one or all child records

- This rule should apply to one or more Constituent Address records
- This rule should apply to all Constituent Address records

When the rule that you are working on is a child record that may have multiple records on the parent, Validatrix needs to know how to handle them. The case above is for constituent addresses but it could equally have been individual phones, action notepads, attributes or volunteer qualifications just to name a few. Note that from version 2.0 patch 2 this can also include top level child objects that are also dependencies such as gifts, actions, participants, etc. When they are not a dependency this setting is ignored.

The easiest way to explain this is with an example.


If my rule is to ensure that the city is in upper case then I would set up the following criterion:



---

Apply when the record agrees with the criterion  
 Apply when the record disagrees with the criterion

Criterion

First letter is upper case  
 Is UPPER CASE  
 Is lower case  
 Validate as an email address  
 Is blank  
 Is not blank  
    
 Is one of...  
  
 Create your own criterion using regular expressions 

The error messages would be shown when the record disagrees with the criterion i.e. the message would be shown if the city is not all in upper case.

However what happens if we have several addresses; one whose city is all in upper case and one that is not?

If we say that we should apply to *all* constituent address records then we show the error message to *any* that disagree with the rule.

So when do we use the other option?

Say we want to check to see that an attribute category has a certain value. This is shown below:




Apply when the record agrees with the criterion  
 Apply when the record disagrees with the criterion

Criterion

First letter is upper case  
 Is UPPER CASE  
 Is lower case  
 Validate as an email address  
 Is blank  
 Is not blank  
    
 Is one of...

Anniversary  
 Special Mailing Types  
 Closing Codes  
 Inactive  
 Origin

Create your own criterion using regular expressions 

`\b(Origin)\b`

In this case if the attribute category is not “Origin” then we will display the error message.

However, again which attribute are we talking about? In this case we do not want the rule to apply to *all* attributes (unless we *only* want “Origin” attributes on a constituent record). In this case we select “This rule should apply to one or more constituent attributes”. That way as soon as the first origin attribute is found we know not to show the error messages. The rule does not look at the other attributes.

### Preferred address or not

There is one option on the overview screen that appears for address record rules only.

This rule only applies to the preferred address (realtime validation)

This option will ensure that only the preferred address is looked at and no other.

### Applying to only specific types of record

For some records it is possible to filter how the rule is applied. In the example below you are able to apply the rule to gift amounts on cash gifts only. The filter is based on gift type. This filter will vary depending on the record type and some record types will not have a filter.



This rule will act on the Amount field (on a Gift)

Rule Name

Description

If set then this rule will only apply to the following gift types

Cash	▼
Cash	▲
Pay-Cash	
MG Pay-Cash	
Pledge	
Stock/Property	
Stock/Property (Sold)	
Pay-Stock/Property	
MG Pay-Stock/Property	▼

Where this comes in real use is when looking at phone types. The filter for a phone record is the format. This is not the phone type but rather how the phone is set up in configuration. It can be a telephone, an email address, a fax number, a web address/ URL or other. If you have set up your configuration correctly in The Raiser’s Edge then all of your email address types (e.g. Email, Email 2, Primary Email, etc.) should all have the format of “Email”. Then you can set up a rule to ensure that all phone types with an email configuration have a valid format.

## Dependencies

### Overview

This is where things start to get trickier!

So far we have looked at rules that work independently of each other. We have rules for a valid email address, for a city that is in upper case letters and for the presence of an attribute category.

However the real power of Validatrix comes in with the use of dependencies. You are able to link Audit Standards together so that a record is not only compared to a single rule but to all the dependant rules too.

This means that you could set up a rule that says the city must be in upper case if the country is in the UK or a rule that says if the constituent has a constituent code of VIP then they must have an attribute of source, a gift amount greater than £1000 but only if their business address is in London, Manchester or Birmingham.



---

In order for a rule to apply i.e. for a message box to be shown, both the rule criteria and the dependency criteria have to apply.

When a rule is used as a dependency no message box is shown for that rule if its criteria agrees or disagrees. Instead, its outcome contributes to the outcome of the parent rule.

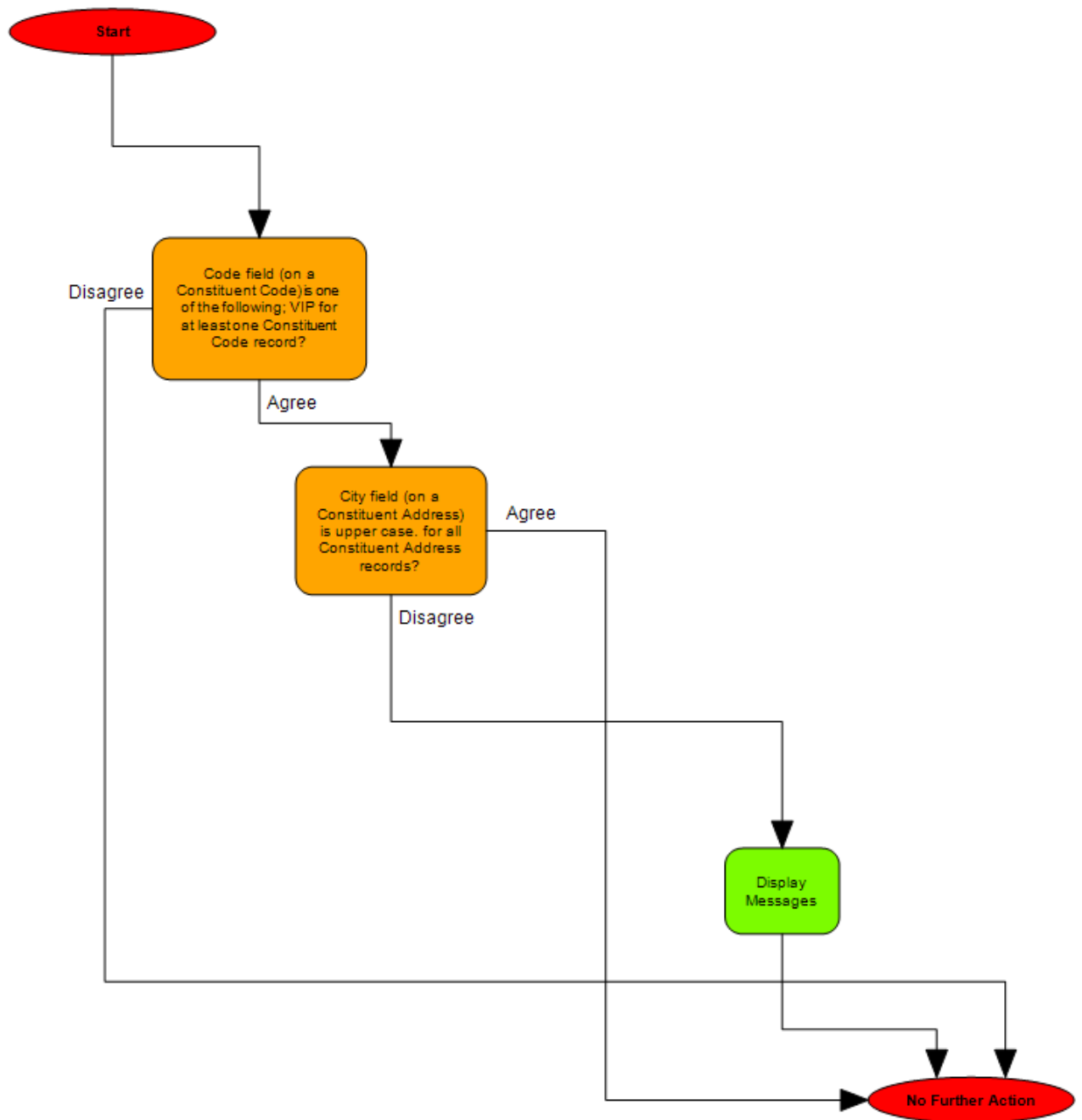
### A simple scenario

We have a rule whose criterion says that the constituent code is VIP. It is marked to show a message box if the record agrees with the rule.

We add a dependency that says the city has to be upper case.

We can look at the flow chart for help as to how we interpret this:





Firstly Validatrix looks to see if the record has a constituent code of VIP. If it does then it looks to see if the city field is upper case. If it does not then the message is shown.

The outcome of the dependency is important when interacting with other dependencies.

### Groups of Dependencies

Dependencies can be grouped together in three ways in order to alter the way they affect their parent rule





---

They can be dependencies where all the rules must be met, dependencies where at least one rule must be met or dependencies where none of the rules must be met. We take a look at each in turn below.

### Dependencies where all rules must be met

The best way to talk about the dependencies is to set up a scenario.


A constituent who has a VIP constituent code must also have EITHER a primary addressee OR a primary salutation value.

We set up three rules; Constituent has a constituent code of VIP, constituent primary addressee is blank and constituent primary salutation is blank. We have already seen the VIP rule and the blank addressee and salutation are identical so here is one of them:

Apply when the record agrees with the criterion  
 Apply when the record disagrees with the criterion

Criterion

First letter is upper case  
 Is UPPER CASE  
 Is lower case  
 Validate as an email address  
 Is blank  
 Is not blank  
 [dropdown]  
 Is one of...  
[text area]

Create your own criterion using regular expressions   
[text input: ^\$]

For our “AND” dependency example we place both the primary salutation and primary addressee rules and dependants on our VIP rule as shown below:



Overview | Rule | Dependencies | Realtime Validation | Audit Standard

All of these rules must be met

Primary Salutation is blank  
 Primary Addressee is blank

At least one of these rules must be met

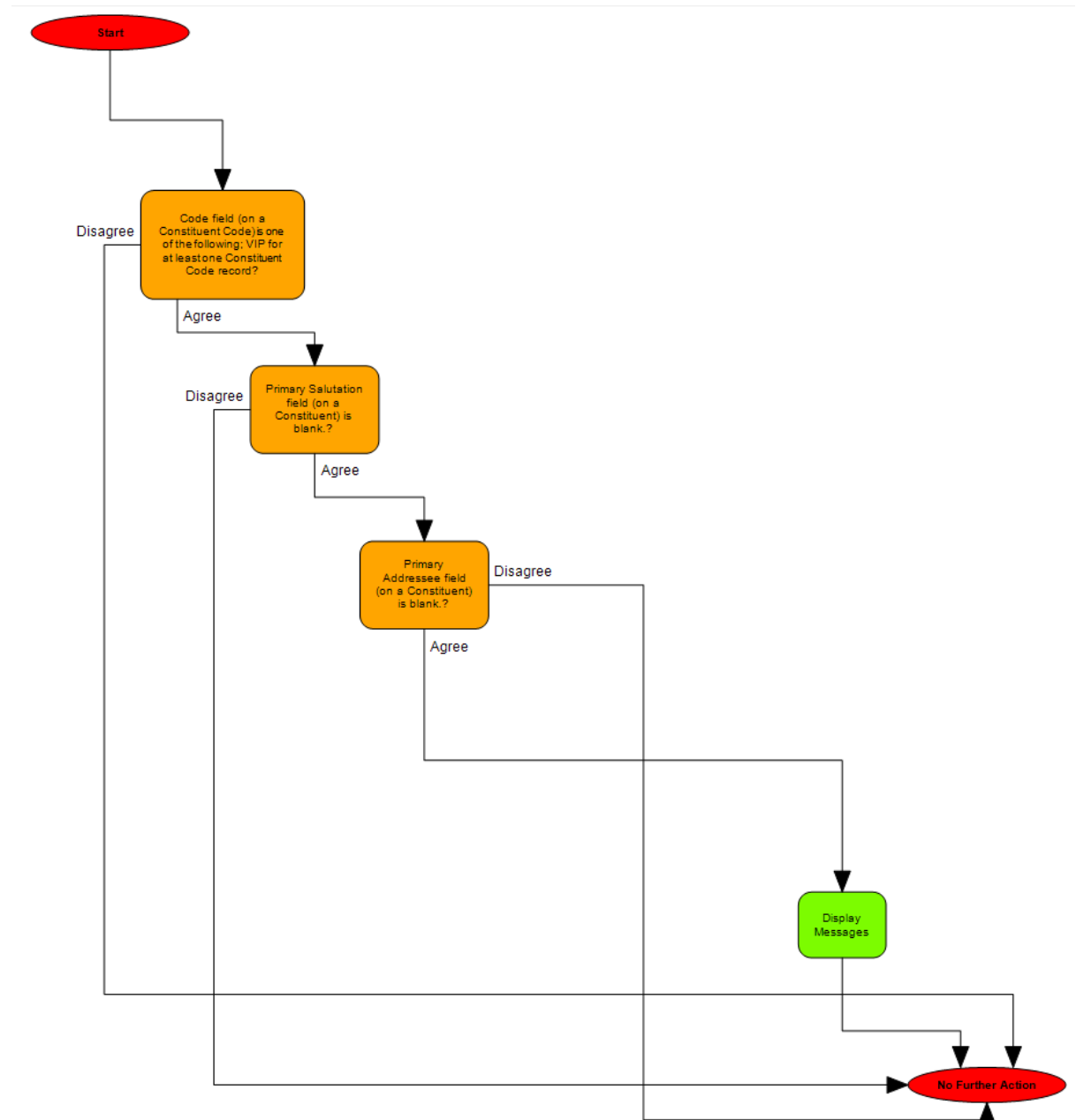
None of these rules must be met

The easiest way of resolving whether a message will be shown for an audit standard is by following the flowchart. However we should also understand what is going on here. The outcome of these dependencies are grouped by a logical “AND”. Assuming that a message would be shown if it were down to the parent rule alone how would the dependencies affect that? The outcomes can be seen below.

Is the primary salutation blank?	Yes	Yes	No	No
Is the primary addressee blank?	Yes	No	Yes	No
<b>Should a message be shown?</b>	Yes	No	No	No

What this says is that Validatrix should only show the message when the primary rule is met i.e. the constituent has a VIP constituent code and both the primary salutation and primary addressee are blank. This is shown below in the flow chart:





### Dependencies where at least one rule must be met

A constituent who has a VIP constituent code must also have BOTH a primary addressee AND a primary salutation value.

This dependency group is a logical “OR”

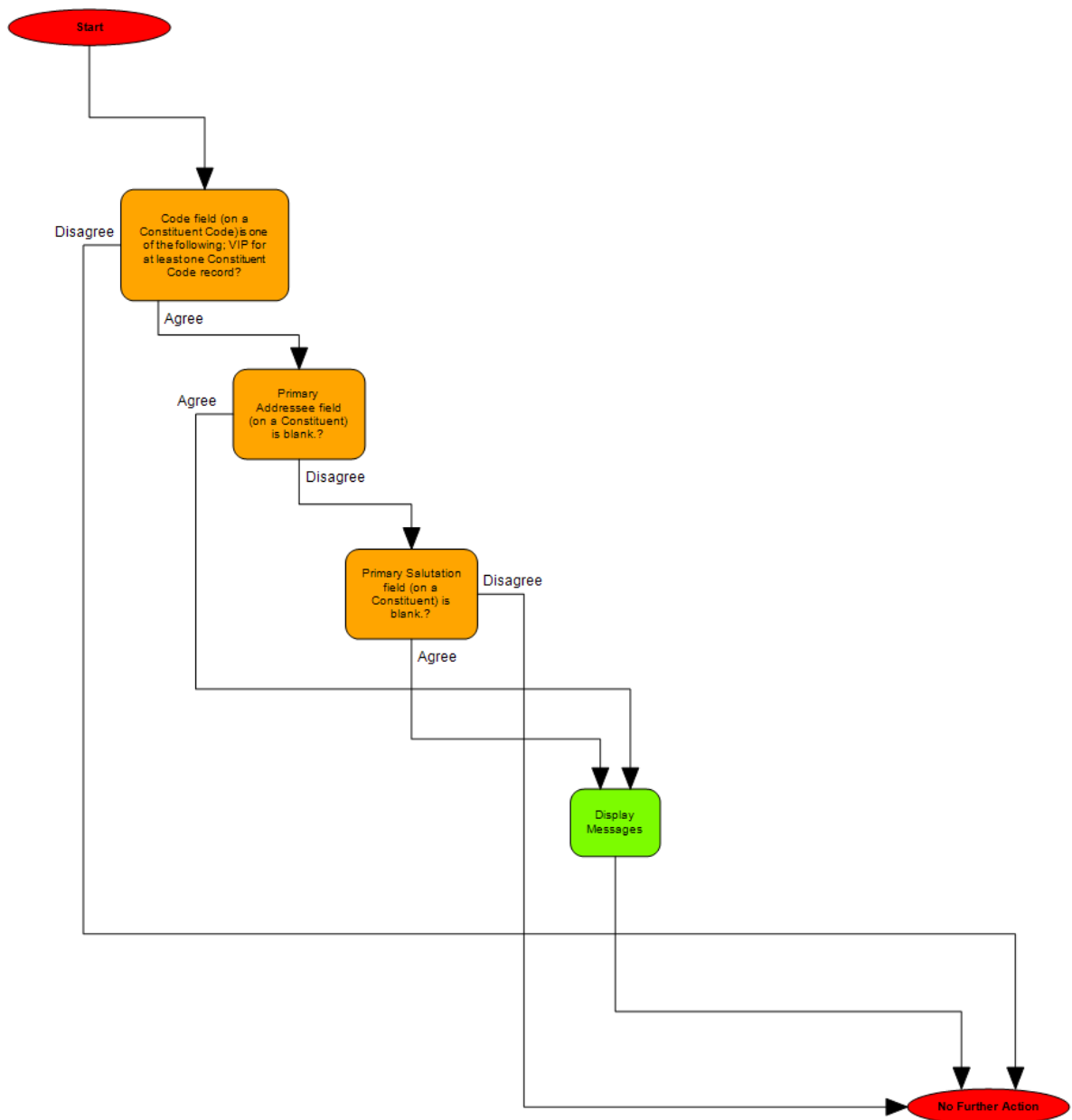
Assuming that a message would be shown if it were down to the parent rule alone how would the dependencies affect that? The outcomes can be seen below.



Is the primary salutation blank?	Yes	Yes	No	No
Is the primary addressee blank?	Yes	No	Yes	No
<b>Should a message be shown?</b>	Yes	Yes	Yes	No

What this says is that Validatrix should only show the message when the primary rule is met i.e. the constituent has a VIP constituent code and either the primary salutation, the primary addressee or both of them are blank.

This is shown below in the flow chart:



---

### Dependencies where no rules must be met

A constituent who has a VIP constituent code must NOT have BOTH a primary addressee AND a primary salutation value.

This dependency group is a logical “NOR”

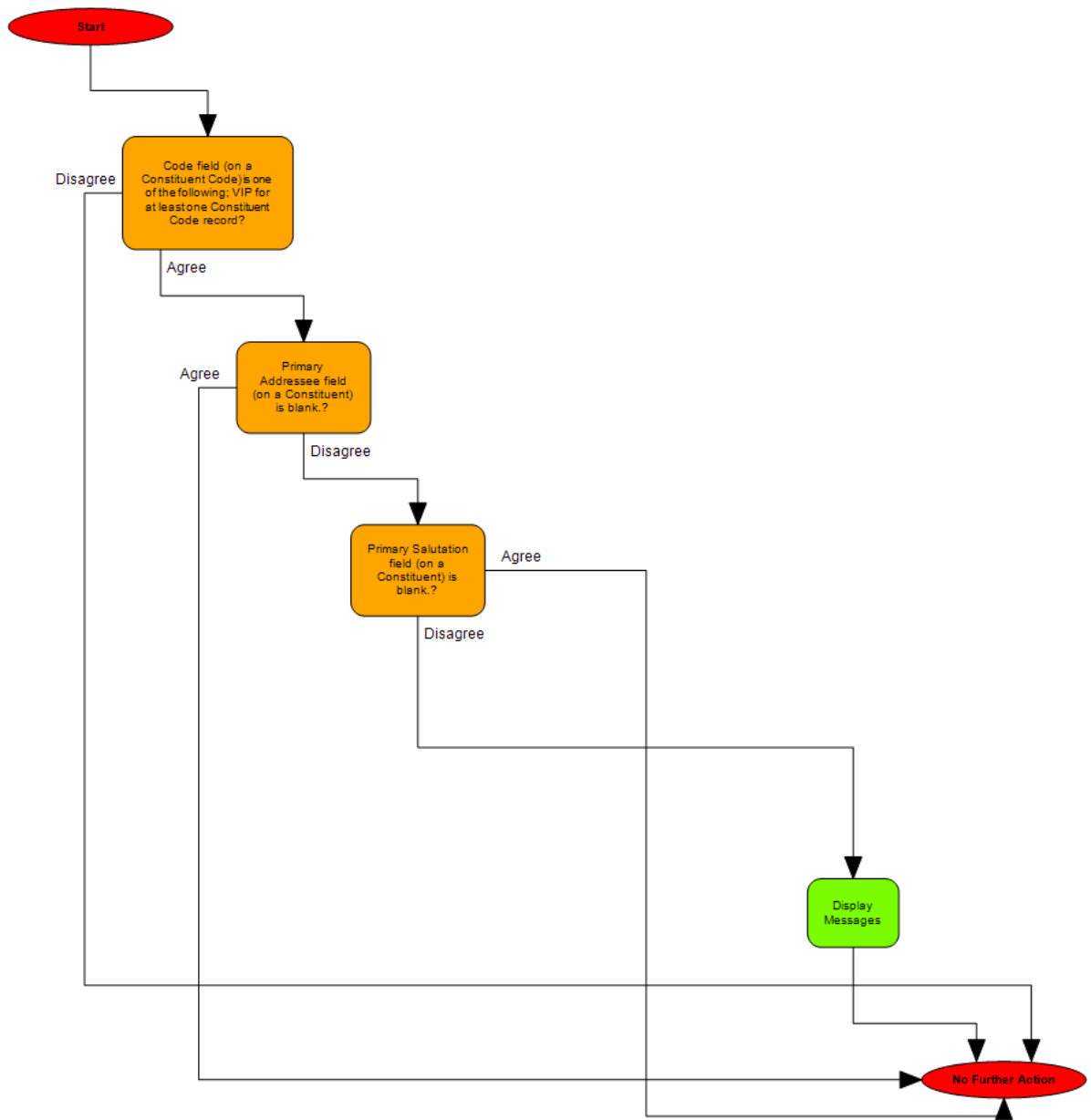
Assuming that a message would be shown if it were down to the parent rule alone, how would the dependencies affect that? The outcomes can be seen below.

Is the primary salutation blank?	Yes	Yes	No	No
Is the primary addressee blank?	Yes	No	Yes	No
<b>Should a message be shown?</b>	No	No	No	Yes

What this says is that Validatrix should only show the message when the primary rule is met i.e. the constituent has a VIP constituent code and both the primary salutation and the primary addressee are blank.

This is shown in the flowchart below:





## A Further Combination of Dependencies

A constituent who has a VIP constituent code should have neither a primary addressee nor a primary salutation value.

There is a fourth combination of dependencies that may be required. There is no box that represents them but the outcome can be seen below:

Is the primary salutation blank?	Yes	Yes	No	No
Is the primary addressee blank?	Yes	No	Yes	No
<b>Should a message be shown?</b>	No	Yes	Yes	Yes



---

What this says is that Validatrix should only show the message when the primary rule is met i.e. the constituent has a VIP constituent code and either the primary salutation or the primary addressee are blank or neither are blank.

This is a logical NAND. If we want this outcome then we have to change the criteria slightly. This is shown below

The screenshot shows a configuration window for Validatrix. At the top, there are two radio buttons: "Apply when the record agrees with the criterion" (unselected) and "Apply when the record disagrees with the criterion" (selected). Below this is a section titled "Criterion" containing several options: "First letter is upper case", "Is UPPER CASE", "Is lower case", "Validate as an email address", "Is blank" (selected), "Is not blank", a dropdown menu with an empty text box, "Is one of..." (with an empty text box below it), and "Create your own criterion using regular expressions" (with a help icon). At the bottom, there is a text input field containing the regular expression `^$`.

In this case the dependencies are placed in the middle box as shown below:



---

All of these rules must be met

+ ×

At least one of these rules must be met

Primary Addressee is blank  
Primary Salutation is blank

+ ×

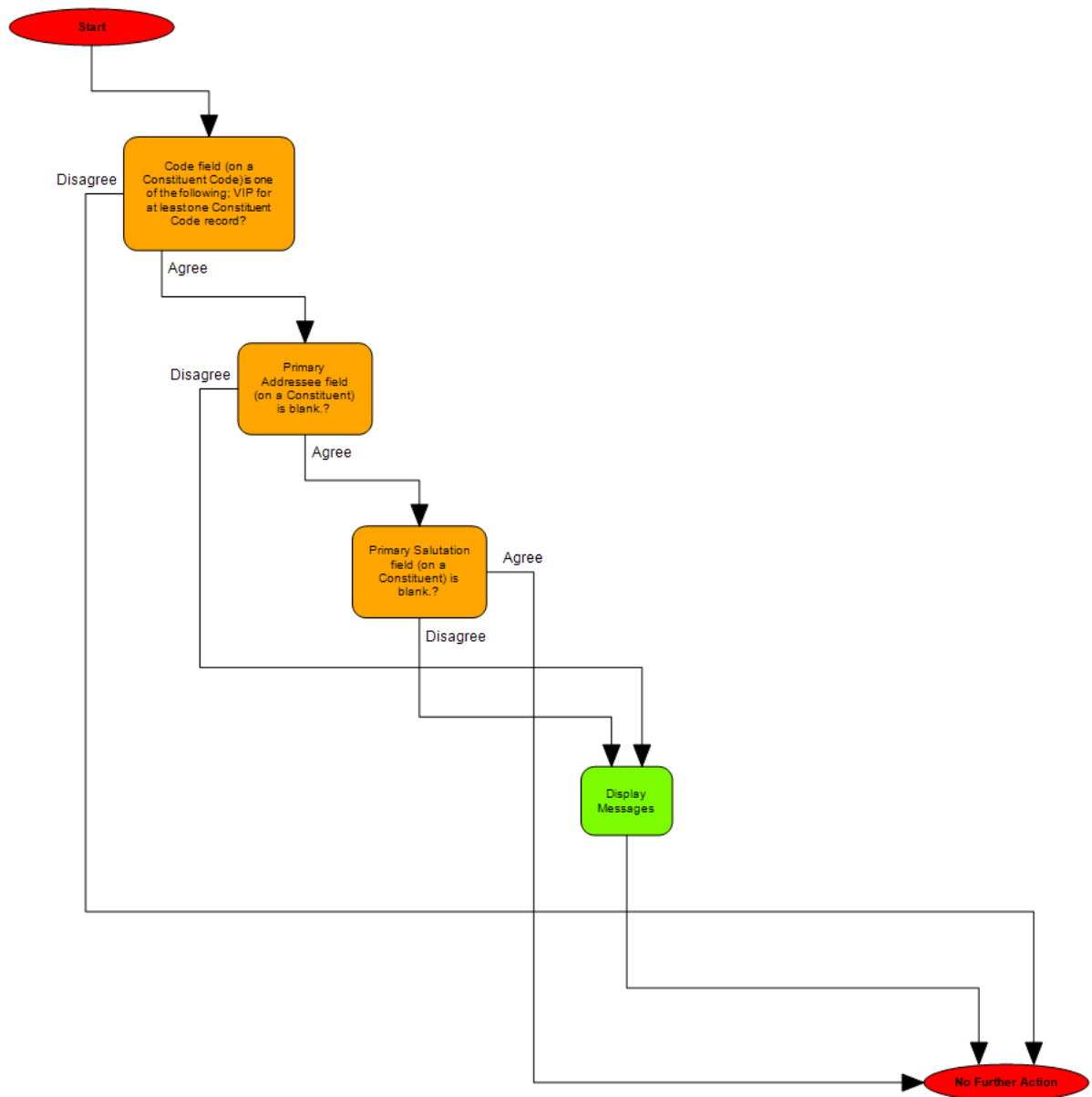
None of these rules must be met

+ ×

This is shown in the flowchart below:







## Avoiding Cross Dependencies of the same type

Imagine the following scenario. My rule needs to check that if I have a constituent code of VIP, I always have a Date From value. This is a relatively simple rule consisting of a parent (checks that the constituent code is VIP) and a dependency (checks that the Date From value is not blank).

However, what happens if I have two constituent codes; one of VIP with a blank Date From and one of Board Member with a value for Date From.

Normally each rule works independently of another which would mean that despite our intention the rule would act.



---

In this case though Validatrix compares the types and the ids of the types so that it takes each constituent code separately and looks at the code and the date from for each.

### An Exception

There is an exception to this behaviour. When the parent rule looks at one field and the dependency looks at the same field on the same record type. In this case the records are not kept together.

In what scenario would you want to look at the same record type and same field but have a different comparison? Here is an example. If I am looking for a specific date on attribute X but only if the constituent does not have attribute Y. In this case we would first look if the constituent has attribute X. Then as a dependency we would have our date check and also a check that they do not have an attribute of category Y.

The check for the attribute category X in the parent followed by the check of lack of attribute Y in the dependency would need to act independently of each other.

## Conclusion

Rules on Validatrix can be extremely powerful. They can also be really quite complicated. In trying to work out why a rule works the way it does the first step is always to look at the flowchart. It will give you the easiest representation of the solution.

Hopefully this guide will be of some assistance to you in developing your own rules. We have added the facility to export and import them from your system so that you can send other organisations your solutions and import theirs.

At the same time we are able to develop rules for you so please get in contact should you require this service.

