CA Agile Requirements Designer: Introduction To The Video Series

This series of videos introduces CA Agile Requirements Designer and how it can be used to model requirements and systems to support collaboration and testing in software projects.

Paul Gerrard

ca® technologies

Models At the Heart of Software Development

Modelling has always been at the heart of software development, but only recently are tools emerging that effectively support modelling, collaboration and testing.

Models Support Thinking

- Modelling and models are the basis of all software design activities.
- We use models everywhere:
  - To represent requirements
  - To represent the behaviour of code
  - To support collaborative design of the product
  - To support the design of the tests of that product
- Many models are never written down; mental models represent our understanding of complex requirements, systems and behaviour.
- CA Agile Requirements Designer supports modelling at all levels.

CA Agile Requirements Designer

- CA Agile Requirements Designer models system behaviours using flowcharts.
- Flowcharts have been around for decades.
- There are many variations of the flowchart-style modelling technique, but all derive from this most fundamental, procedural model type.
- If you understand flowcharts:
  - You are modelling when you use a flowchart to capture a requirement or procedure.
  - You know how modelling helps in your day to day activities.
  - You will find CA Agile Requirements Designer easy to use.
- In these videos, you will learn how to think with models and how to apply CA Agile Requirements Designer to a range of common situations.

Thinking With Models

- The New Model for Testing is a proposed framework for thinking like a tester.
- It is appropriate for users who test, and testers and developers when they are testing.
- The New Model sets out 10 thinking activities that exist in every test process.
- We'll explore how models are used to capture and validate requirements and how to derive tests from those same models.
- Tools and CA Agile Requirements Designer in particular must be used in context, so we offer a modelling process that complements your development activities.

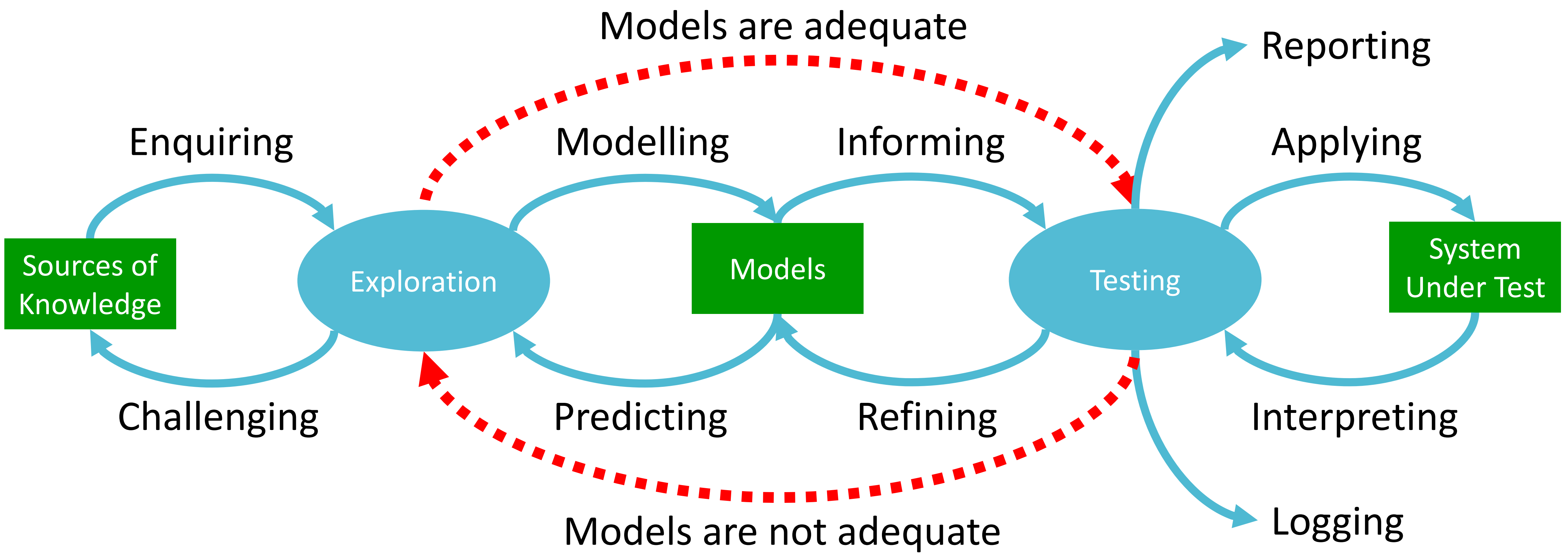
Modelling Requirements and Testing

- A range of videos will demonstrate how CA Agile Requirements Designer can be used to model:
  - Decision tables
  - Graphical user interfaces (GUIs)
  - Complex, end-to-end processes and user journeys that depend on system features
  - Extract, transform and load processes
  - Loops and repetition in systems
  - RESTful web services
- Other videos will demonstrate selected features of the tool:
  - Creating data and expected results
  - Generating paths or test cases using a range of optimization or coverage techniques
  - The all-pairs test approach
  - The use of filters to select model element subsets and navigate complex models

Summary

- CA Agile Requirements Designer has a large number of useful features.
- We will only explore the core functions relating to model design, path optimization, modularisation, data and expected results.
  - CA Test Data Manager is out of scope for these videos, but is a very powerful tool to find, make and manage data and expected results.
  - CA Service Virtualization is also out of scope–it provides a comprehensive solution to the problem of testing sub-systems in isolation or where service components are not yet available.
- To date, testers have had a wealth of automated tools that support mechanical activities like test execution.
- CA Agile Requirements Designer is one of the first tools that truly supports tester thinking and design activity.

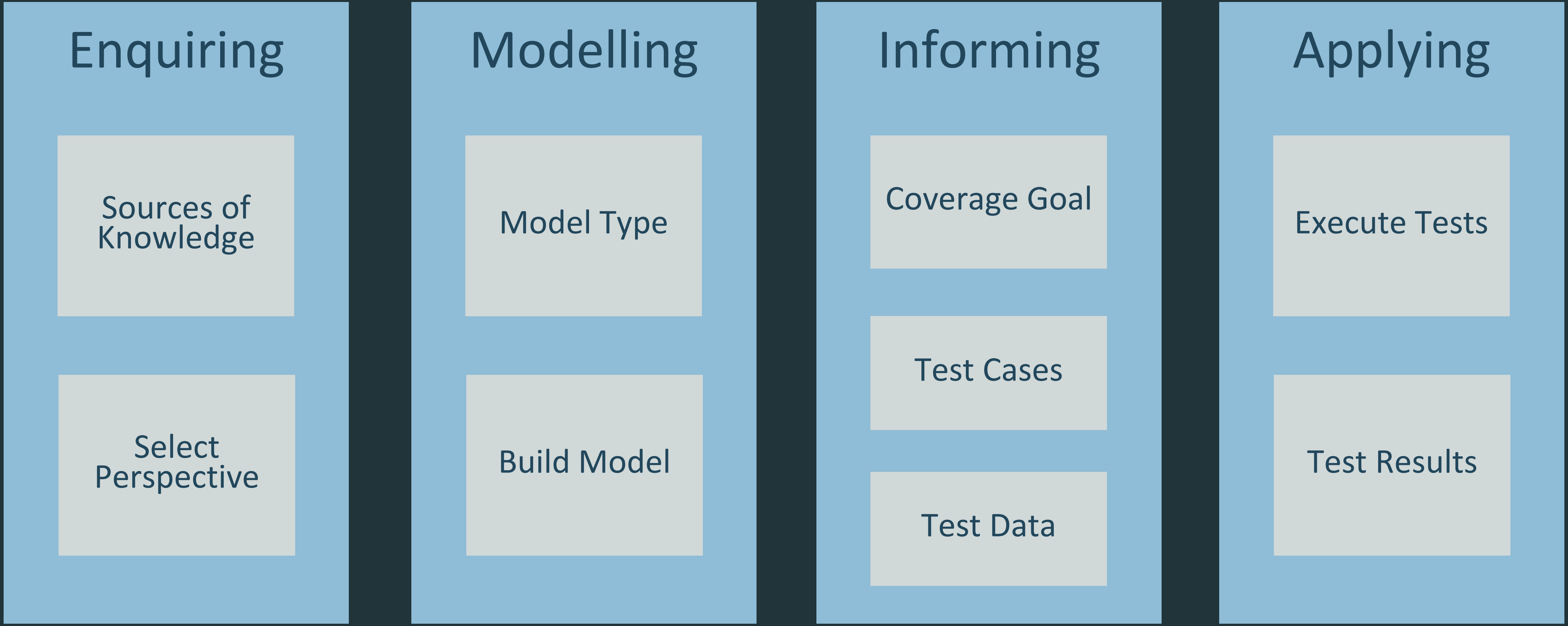
New Model Testing



Shared Understanding and Collaboration

- The New Model suggests that the thought process of:
  - The Developer (who tests) and
  - The Tester
- …are the SAME.
- Collaboration between developer, tester and user/product owner is a common pattern.
- Shared model(s) form the basis of shared understanding and collaboration.

The CA Agile Requirements Designer process



Enquiring

- Sources of Knowledge
- Select Perspective

Modelling

- Model Type
- Build Model

Informing

- Coverage Goal
- Test Cases
- Test Data

Applying

- Execute Tests
- Test Results

Testing GUIs - Our Strategy for the Insurance Premium Calculator

- The five sections of the form can't really be tested independently of each other—they are all part of the same form in reality.
- If we use subprocesses in our models, when we perform the optimizations on parent models, the default subprocess integration is called stored paths.
- Our strategy for path optimization will be:
  - For each subprocess, select an optimization for all start and endpoints, and store the paths.
  - Because most paths are terminated when an error is encountered, we'll attempt all paths optimization to start with.
  - For models with loops, we'll use one single iteration for the loops.
  - If the number of paths is excessive, we'll use all edges instead.
  - Because of the shape of most models, all edges generates a similar number of paths to all pairs and all in-out edges.

How to Create a Model for a GUI Form

1. The first thing to try is to submit the form with no entered data; this should flag up the mandatory data fields required, for example:
   - Data fields that must have a non-null value
   - Drop-downs that must be selected
   - Radio buttons and check-boxes that must be clicked
   - And so on
2. Select drop-downs, check-boxes and radio buttons and identify those that trigger elements on the web page to appear or disappear.
   - Select options, and without adding data, try to submit the form.
3. Look for validation rules in fields, e.g. string length, dates, number ranges etc.
4. Consult the specification, users and/or developers to complete your analysis.

Testing Web Services

- CA Agile Requirements Designer can be used to model web services.
- We used a simple RESTful service to demonstrate, but in principle, all web services can be tested in a similar way.
- Of course, you would use either a dedicated services test tool or a custom-built driver to programmatically test your own web services—the technicalities differ.
- You can see that the order of parameters, however supplied, doesn't matter.
- If you know the validation and processing rules, you can create tests that cover both the input validation and processing rules.