Verification & Validation

UPSSITECH

October 2023

Verification & Validation

- **Verification**
  - “Are we building the system right ?”
  - The system implementation meets its specification and exhibits desired properties

- **Validation**
  - “Are we building the right system ?”
  - The system meets the needs and expectations of stakeholders

Verification & Validation

- V&V must be applied at each stage of the development process
  - Detect defects earlier
  - Emphasize quality throughout development
  - Assess whether or not the system is usable in an operational situation

- V&V should confirm that the system is fit for purpose
  - This does NOT mean completely free of defects BUT it must be good enough for its intended use

V&V using Ingescape

- Domain Specific Language (DSL) to write V&V tests:
  - Instructions to inject information and events in a given platform
  - Verifications to check that the behaviors of agents is correct
  - Documentation available at https://ingescape.com/verification-validation-using-ingescape/

- Tool dedicated to running V&V tests on Ingescape platforms and generating reports

V&V using Ingescape: Instructions

Instruction	Code
Inject information on the input of an agent	myAgent.myImpulsion = 0 // 0 by convention but any value is valid myAgent.myInt = 345 myAgent.myBool = 1 // 0 means false and 1 means true myAgent.myDouble = 23.56 myAgent.myString = "my string" myAgent.myData = "0xaa22eeff" // hexadecimal binary data
Call a service	myAgent.myService(1, 23, 45.213, ”mystring", "aa32eed4")
Sleep	sleep 1000 // Wait for 1000 ms
Interrogate a user through the command line interface	ask ”Q1 without text" ask ”Q2 with a specific character from the following list" y n ? ask ”Q3 with free text" text

V&V using Ingescape: verifications

Assertion	Code
Output value	assert myAgent.myDoubleOutput >= 10.234 assert myAgent.myIntOutput == 42 assert myAgent.myStringOutput == "expected message with value = 45" assert myAgent.myStringOutput ~ "expected [^]+ with value = (\d+)"
Silence	assert silence myAgent 5000 //all the outputs of myAgent assert silence myAgent.myOutput 5000 //silence on myOutput
Service	assert getAnswer from myAgent // we received the call assert getAnswer from myAgent with arg1 = 1 // checking first arg assert getAnswer from myAgent with arg2 = 45 and arg4 = "Hello" // checking args 2 and 4 assert getAnswer from myAgent with arg4 = "~ str(.*)"
Interrogate a user (y or n)	assert user "are you ready ?"
Logs	expect log 15000 myAgent DEBUG == "model_write_iop;set input myDouble to double 13.890000" expect log 5000 myAgent DEBUG ~ "model_write_iop;set output value1 to (.*)"

V&V using Ingescape: blocks

	Code

Blocks	{ // Untitled block
	INSTRUCTIONS_AND_VERIFICATIONS
	}
	
	"my title" { // With a title only
	INSTRUCTIONS_AND_VERIFICATIONS
	}
	
	"my title" "my description" { // With a title and a description
	INSTRUCTIONS_AND_VERIFICATIONS
	}
	
	"my title" "my description" {
	// Run multiple verifications at once based on a given event
	block.use_pool = true
	block.multi_check = true
	INSTRUCTIONS_AND_VERIFICATIONS
	}

V&V using Ingescape: running V&V tests

- macOs:

```
cd /Applications/Ingescape/Ingescape-Circle.app/Contents/agents/igs

./igs --device MY_DEVICE --port MY_PORT --script MY_SCRIPT_FILE
```

- Windows

```
Go to Circle install directory (Program files)

igs.exe --device MY_DEVICE --port MY_PORT --script MY_SCRIPT_FILE
```

- Linux

```
cd /opt/Ingescape-Circle/bin

./igs --device MY_DEVICE --port MY_PORT --script MY_SCRIPT_FILE
```

V&V with Ingescape: example

- Whiteboard app
  - V&V script (*Whiteboard.igsscript*) available in the open repository for the Whiteboard agent
  - https://gitlab.ingescape.com/learn/whiteboard/-/blob/main/Whiteboard.igsscript