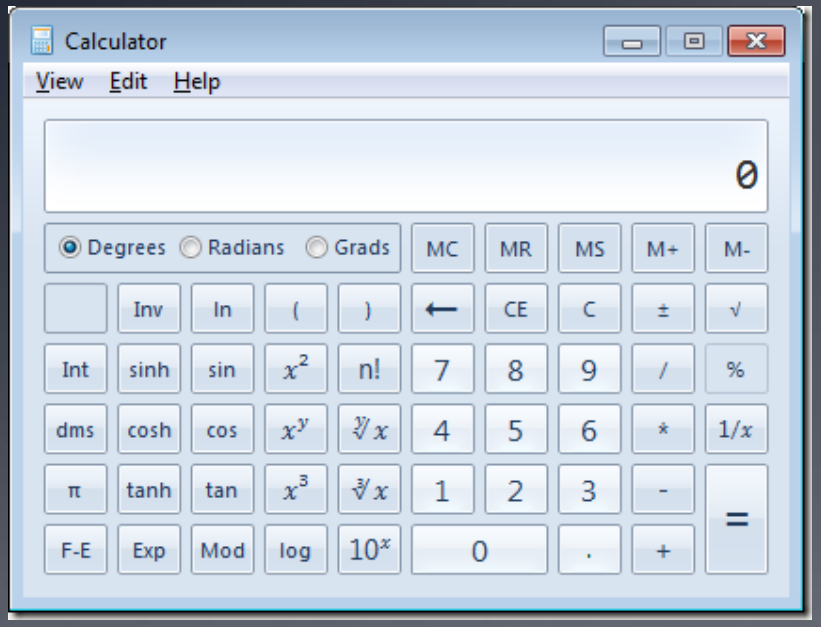
Visuell GUI Testning

Automatiserad System- och Acceptanstestning

2012-10-3

Michel Nass, Inceptive
Emil Alégroth, Chalmers

Vad är ett Graphical User Interface (GUI)?



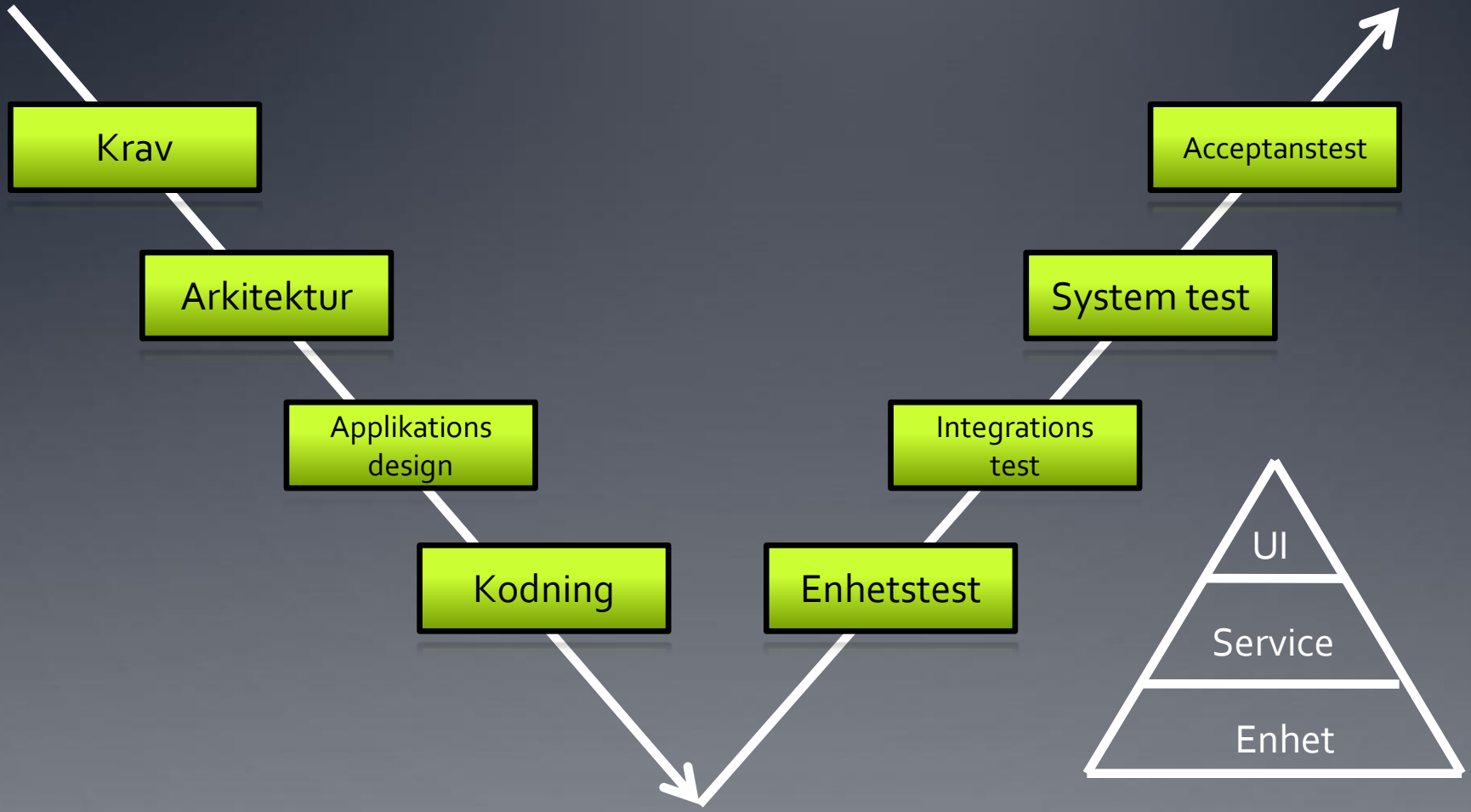
Icke-animerat GUI

Animerat GUI

Nuläget

- System- och acceptanstestning är dyrt!
  - Manuellt
  - Långsamt
  - Enformigt
  - Svårt att replikera exakt
  - Nödvändigt
- Regressionstestning
  - Verifiera att systemet fungerar efter underhåll/nyutveckling
  - Validering av systemet med kund-feedback

Vad är svårast att Automatisera?



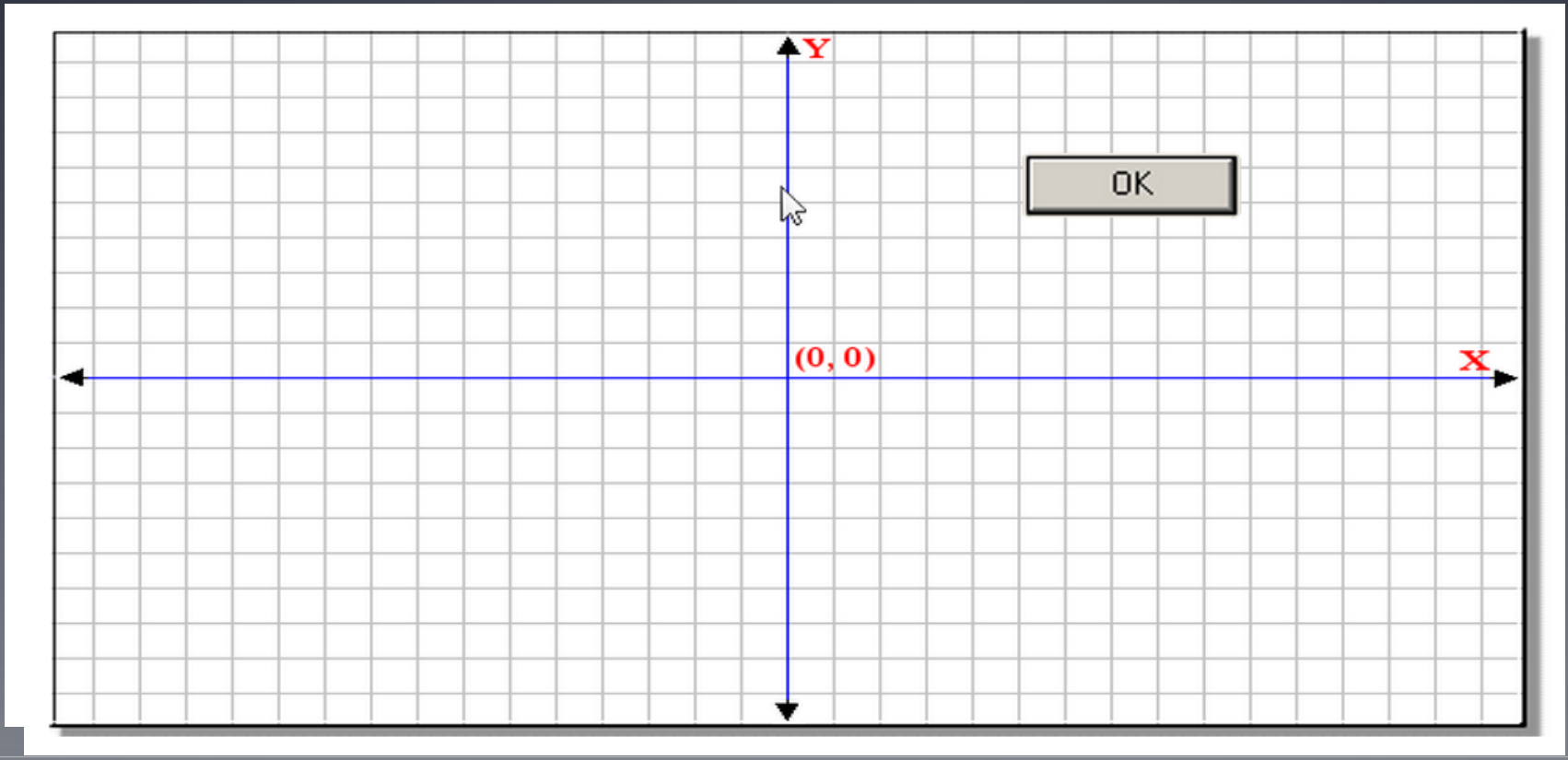
System nivåer

Tidigare tekniker

- GUI baserad testning
  - Testning via GUI:t, inte av GUI:t
- Funnits verktyg i många år
  - TestComplete, HP Quick Test Professional, IBM Rational Functional Tester, Visual Studio Test Professional, etc.
- Använts i industrin
  - Med tveksamt resultat
- Två grupper av tekniker
  - Första generationen: Koordinatbaserad testning
  - Andra generationen: Widgetbaserad testning

Första generationens testning

- Den första generationens testautomatiseringsverktyg baserades på absoluta koordinater

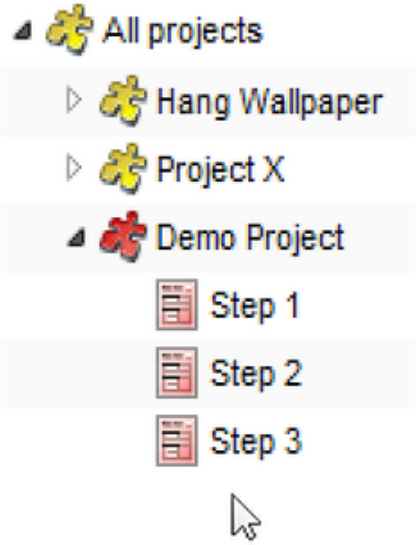


För- och nackdelar

- Enkel teknik som är lätt att lära och förstå (Inlärningsbarhet)
- Fungerar inte bra för automatisering av fönsterbaserade applikationer

Andra generationens testning

- Den andra generationens testautomseringsverktyg baseras på identifikation av Widgets



Name	Tree1
X	100
Y	120
Width	450
Height	500

```
Tree1.MouseMove(100, 300)
LeftMouseClick
```

För- och nackdelar

- Okänslig för ändring av fönsterstorlek
- Widget:en behöver ej vara synlig på skärmen

- Använder första generationens teknik för Widgets som inte stöds av verktyget
- Stora och dyra verktyg
- Komplex hantering av komponentdatabas
- Uppspelning görs inte på exakt samma sätt som när man gör det manuellt
- Stödjer bara ett begränsat antal typer av gränssnitt och miljöer
- Tvingas utvärdera verktygets testbarhet för applikationen man avser att testa
- Måste samarbeta med utvecklare för att bl.a. Bygga in identiteter i de grafiska objekten

Visuell GUI Testning (VGT)

- Teknik med verktygsstöd för automatiserad System- och Acceptanstestning
  - Bildigenkänning
  - Scenario-baserad testning
- Möjliggör emulering av en mänsklig användare
  - Artificiell syn
  - Simulering av mus och tangentbord
- Verktyg
  - JAutomate, Sikuli, EggPlant, SeeTest, T-Plan Robot, Testing Anywhere

Pseudokod exempel

MouseClick()

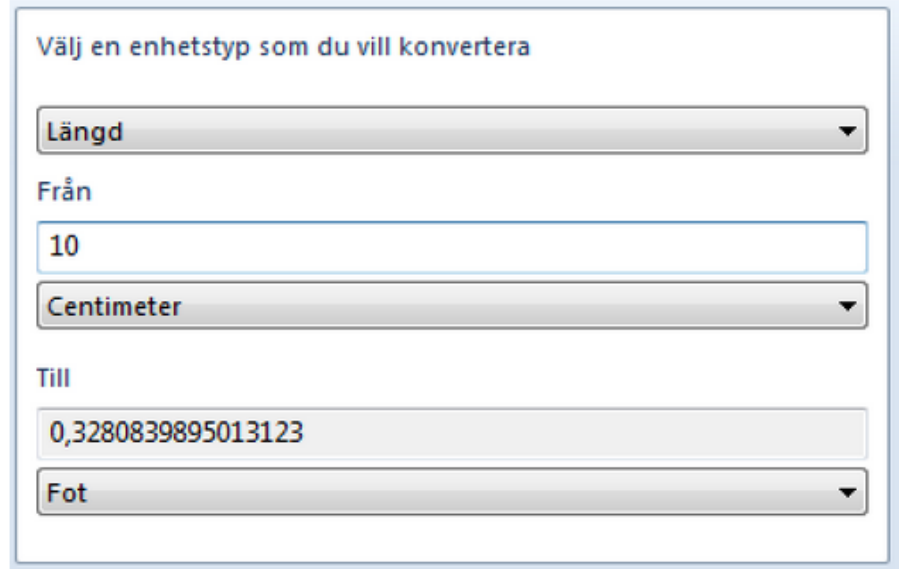
Write(”admin \t admin”)

MouseClick(

)

Wait(”You are logged in”)

Exempel




```
Click( Ange värde )

Write("10\n")

Verify( 0,3280839895013123 )
```

JAutomate

- Automatisk inspelning av testfall
- Kan söka fram dolda bilder och texter
- Manuella och halvautomatiska teststeg
- Stöd för datadrivna testfall
- Exporterar testfall till FitNesse

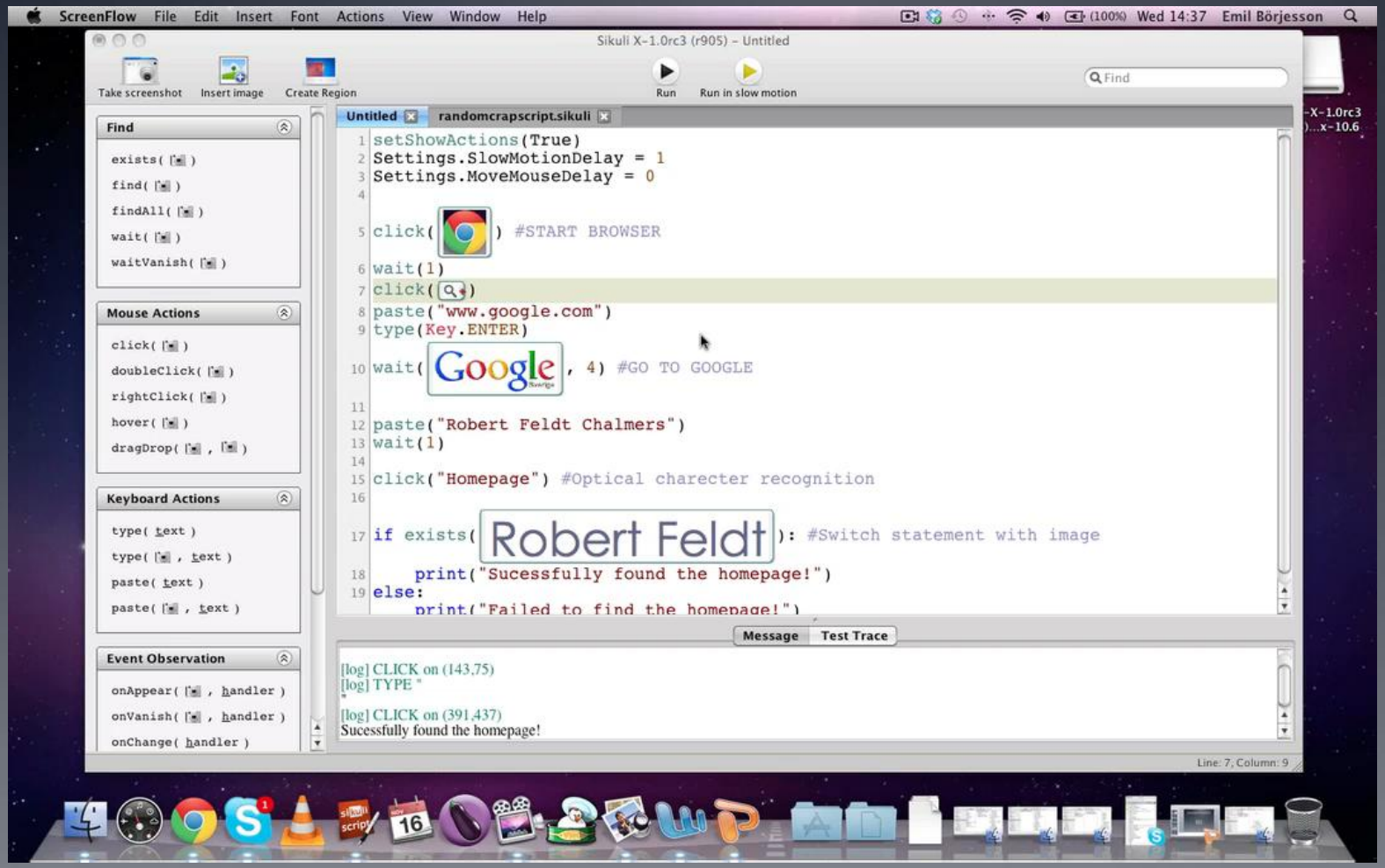
Demo

- Live demo av JAutomate

Sikuli

- Open source (Utvecklat på MIT)
- Python som skriptspråk
- Tillåter importer av Python och även Java bibliotek
- Optical character recognition (OCR) stöd
- Exportering av körbara skript

Video


ScreenFlow File Edit Insert Font Actions View Window Help
Wed 14:37 Emil Börjesson
Sikuli X-1.0rc3 (r905) - Untitled
Take screenshot
Insert image
Create Region
Run
Run in slow motion
Find
Untitled
randomcrapscript.sikuli
setShowActions(True)
Settings.SlowMotionDelay = 1
Settings.MoveMouseDelay = 0
click( ) #START BROWSER
wait(1)
click( )
paste("www.google.com")
type(Key.ENTER)
wait(, 4) #GO TO GOOGLE
paste("Robert Feldt Chalmers")
wait(1)
click("Homepage") #Optical charecter recognition
if exists( ): #Switch statement with image
print("Sucessfully found the homepage!")
else:
print("Failed to find the homepage!")
Find
exists( )
find( )
findAll( )
wait( )
waitVanish( )
Mouse Actions
click( )
doubleClick( )
rightClick( )
hover( )
dragDrop(,)
Keyboard Actions
type(text)
type(, text)
paste(text)
paste(, text)
Event Observation
onAppear(, handler)
onVanish(, handler)
onChange(handler)
Message
Test Trace
[log] CLICK on (143,75)
[log] TYPE "
"
[log] CLICK on (391,437)
Sucessfully found the homepage!
Line: 7, Column: 9

Forskningsstudie

- Forskningsfråga: Fungerar VGT i industrin?
  - Jämförelse mellan Sikuli och CommercialTool
- Saab ATM
  - Säkerhetskritiska flygledningssystem
- 2 delar
  - Akademiska experiment
    - Fyra leksaks-exempel
  - Industriell evaluering
    - Automation av 10 % av en manuell testsuite

Experiment Studien

- Experiment 1 (icke-animerat GUI)
  - Interaktion med miniräknare (beräkna 6+9)
  - Commercial tool: 100 %
  - Sikuli: 50 %

- Experiment 2 (icke-animerat GUI)
  - Identifiera en liten grafisk förändring på en stor yta
  - Commercial tool: 100 %
  - Sikuli: 100 %

Experiment Studien

- Experiment 3 (animerat GUI)
  - Identifiera en given frame ur ett video-clip
  - Commercial tool: 3 %
  - Sikuli: 25 %

- Experiment 4 (animerat GUI)
  - Följ ett objekt över en multifärgad yta
  - Commercial tool: 0 %
  - Sikuli: 100 %

Testfall Exempel (Scenario)

Input	**Förväntat resultat**	**(Ja/Nej)**
Klicka på knapp X	Knapp X byter färg från grönt till rött	Ja
Klicka på knapp Y	Knapp Y byter från grön till röd färg, knapp X byter från grön till röd färg.	Ja
Skriv "ABCD" I textfält A och klicka på knapp Z.	Knapp Z byter från rött till grönt. Texten "ABCD" visas i Text-label B.	Nej

Results

	Commercial Tool			Sikuli			
Testfall	Dev-time (min)	Exe-time (sec)	LOC	Dev-time (min)	Exe-time (sec)	LOC	TC Rows
1	255	111	103	105	90	212	5
2	195	405	233	200	390	228	4
3	285	390	368	260	338	345	13/16
4	205	80	80	180	110	92	9
5	120	90	115	150	154	169	8
Totalt:	**~18 timmar**	**~17 minuter**	**899 LOC**	**~18 timmar**	**~17 minuter**	**1046 LOC**	

Resultat

- Högre applicerbarhet med Sikuli (Fler GUI typer)
- Antal scriptbara testfall: 98%
- Snabbare testkörning (est. för 50 testfall): 5.3 ggr snabbare
- Estimerad utvecklingstid (50 testfall): 20 affärsdagar
- Verktygen lika bra (Ingen statistiskt signifikant skillnad)

Ytterligare forskning

- Acceptans-studie på Saab Göteborg
- Introduktions-studie på Saab Järfälla
  - Underhållskostnader: 25% av utvecklingskostnaden
  - Positiv ROI av utvecklingskostnaden efter 12 exekveringar
- Random Visuell GUI Testning
- Robusthets-analys vid Saab Göteborg

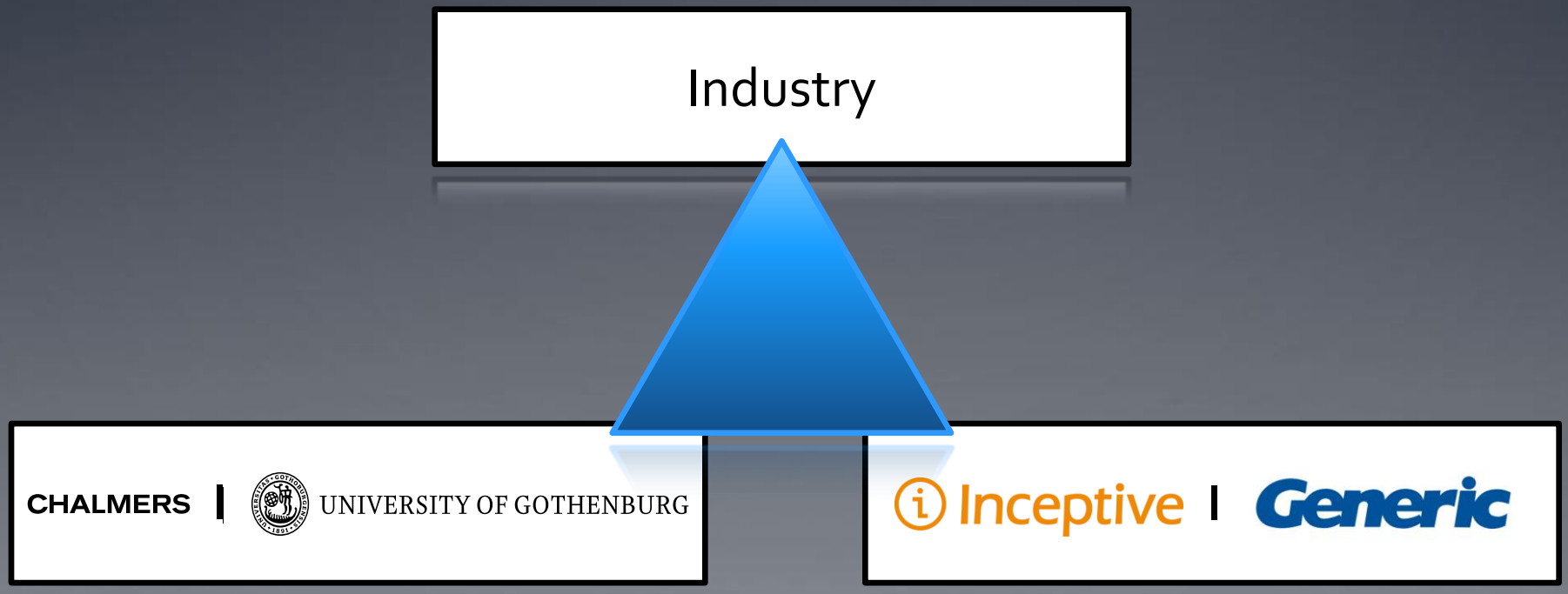
Fördelar och utmaningar

- Oberoende av bakomliggande teknik
- Testar på samma sätt som en mänsklig testare
- Enkelt att förstå och ändra vad testfallet gör
- Kräver låg teknisk kompetens

- Införande-kostnader
- Begränsas av SUT:ens mognad och hastighet
- Verifierade underhållskostnader

Samarbete är nyckeln

- Vad behövs för att förbättra VGT?
  - Fler fall – Olika system och domäner
  - Mer samarbete – Industriell öppenhet
  - Vi måste göra detta tillsammans!



Frågor

- Tror ni VGT kommer lösa de problem ni har idag?