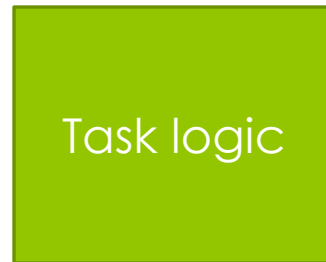
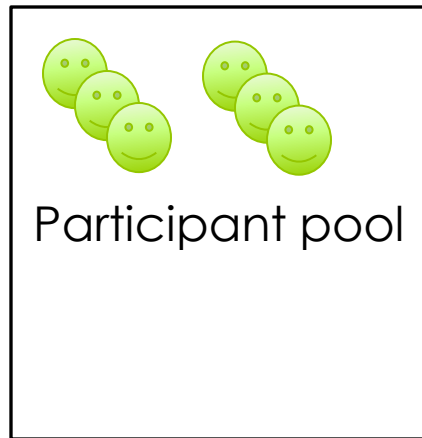




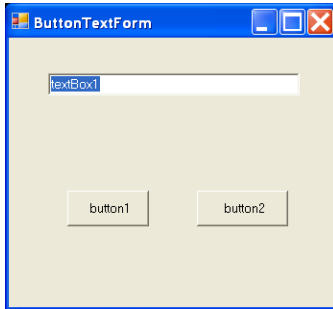
Standard Task-Actor Protocol

Vladislav “Dan” Veksler

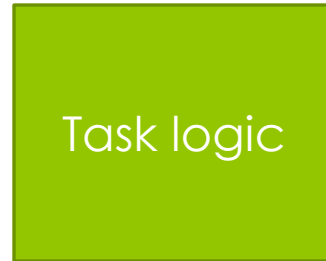
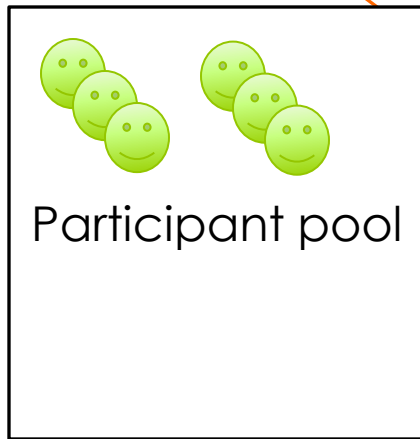




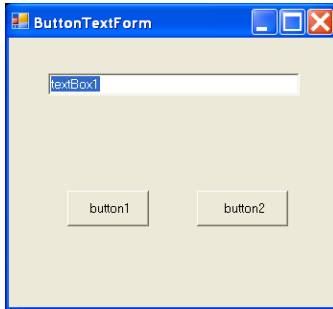
Task ↔ Actor



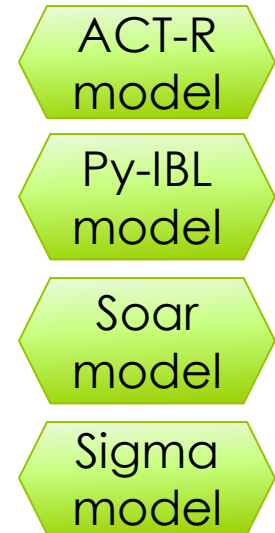
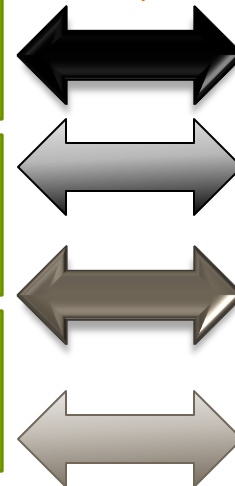
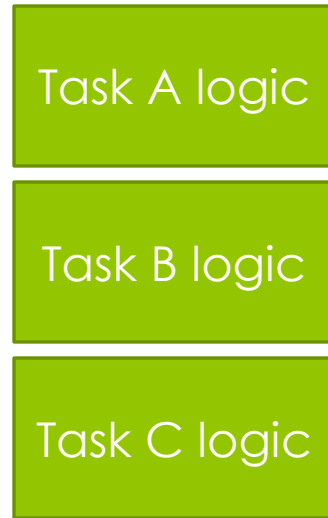
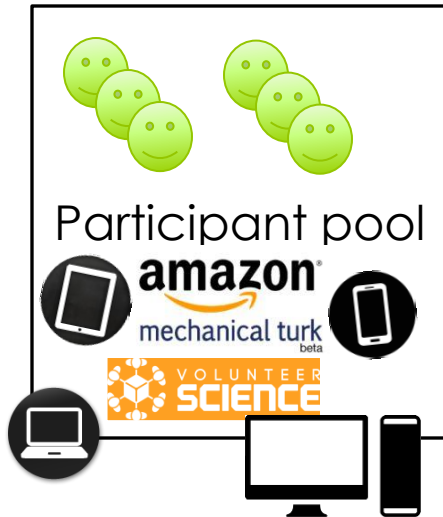
Loc	Att	Kind	Value	Color	ID
(25 25)	NEW	TEXT	"textBox1"	BLACK	VISUAL-LOCATION0
(35 55)	NEW	BUTTON	"button1"	BLACK	VISUAL-LOCATION1
(55 55)	NEW	BUTTON	"button2"	BLACK	VISUAL-LOCATION2



Tasks ↔ Actors



Loc	Att	Kind	Value	Color	ID
(25 25)	NEW	TEXT	"textBox1"	BLACK	VISUAL-LOCATION0
(35 55)	NEW	BUTTON	"button1"	BLACK	VISUAL-LOCATION1
(55 55)	NEW	BUTTON	"button2"	BLACK	VISUAL-LOCATION2



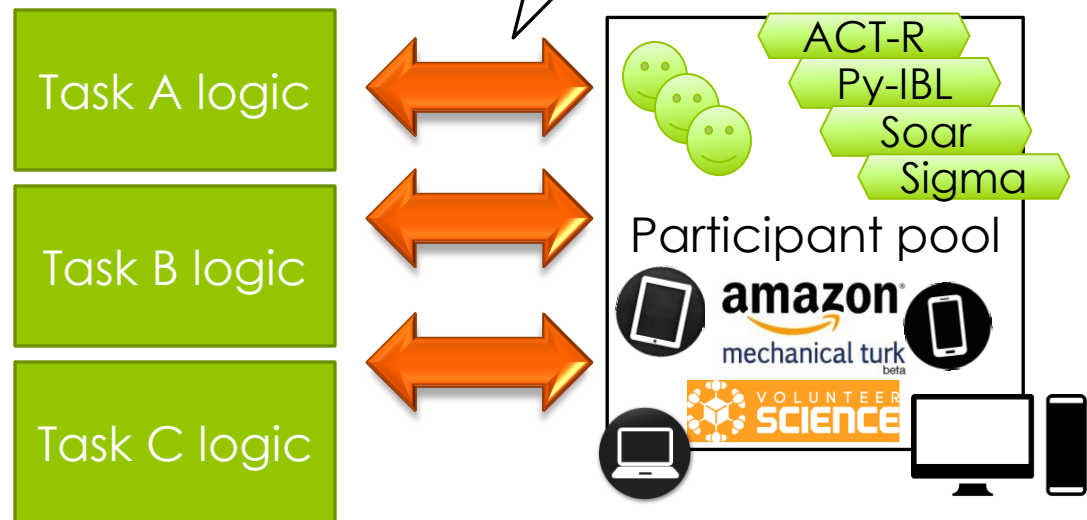
Does the model scale to other tasks? Does the task scale to other models?

- How much work is needed to interface a given model with a new task-environment?
- Is a given task-environment suitable for cross-framework modeling simulations, as well as human empirical studies?
 - Will different models and human actors all be presented with similar information?

Creating a standard task-actor protocol

- Save **R**esources
- Task/model **R**euse
- Scientific **R**eplication

* **R**estriction



STAP (Simple Task-Actor Protocol)

- Worked on as a part of Robotics and Network Science Collaborative Task Alliances



<https://github.com/vdv7/stap>

Demo

	Shepard, Hovland, Jenkins (1961)	IED-tactical (hearts & minds)	SS-RICS robot navigation task
Human Participant	✓	✓	✓
ACT-R (lisp)	✓	✓	✓
IBL (python)	✓	✓	✓

STAP v1.0

```
{  
  <<handshake>>  
  "s": <<state>>,  
  "t": <<title info>>,  
  "a": <<allowed actions>>,  
  "r": <<reward>>,  
  "$": <<score>>  
}
```

- Meant to allow
 - Hierarchical state description and vector graphics
 - Varying types of actions (e.g., click, hold-down, type)
 - Varying types of feedback (e.g. success/fail, temp reward, long-term score)
 - FTRT simulations
 - Seamless playback & 3rd party observation
 - Backwards compatibility
 - Auto-genera instructions

STAP v1.0

```
{  
  "s": <<state>>,  
  "t": <<title info>>,  
  "a": <<allowed actions>>,  
  "r": <<reward>>,  
  "$": <<score>>  
}
```

- **R**esource savings, **R**euse, **R**eplication
 - Don't worry about gui development (just pick a template)
 - Write task logic once, serve it to varying devices and computational models
- Standard logging/playback
- Online studies or FTRT local simulations



Questions?

```
{  
  "s": <<state>>,  
  "t": <<title info>>,  
  "a": <<allowed actions>>,  
  "r": <<reward>>,  
  "$": <<score>>  
}
```

