



# “SPRING to Life”

## Developing Simulations in the SPRING Framework

Craig Cornelius, Ph.D.

SUMMIT

Stanford University School of Medicine

## Outline

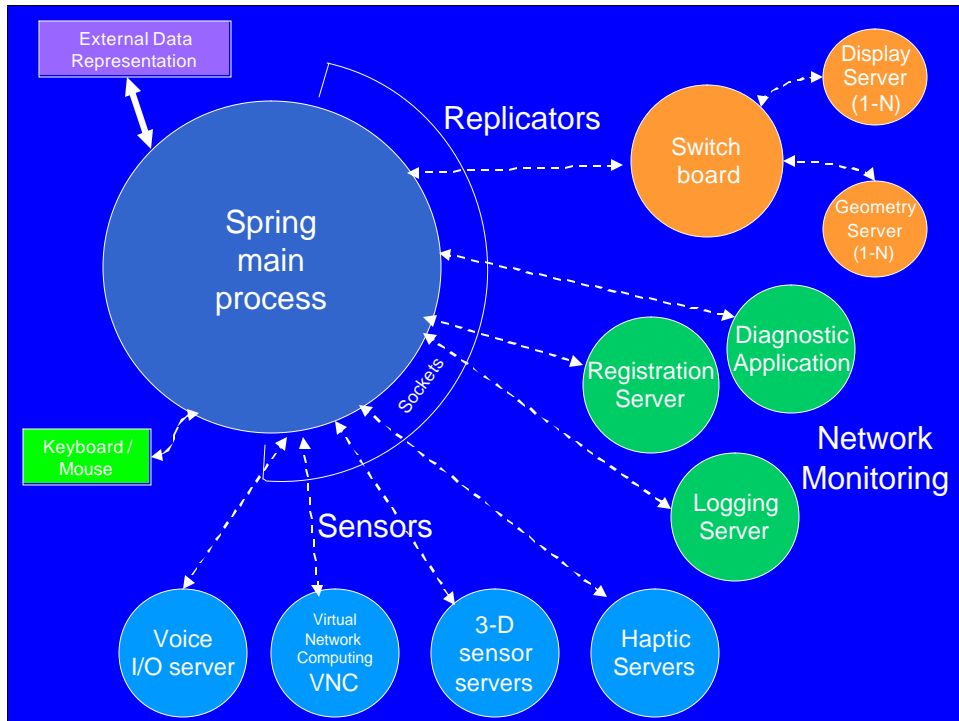
- What’s already built into SPRING
- Configuring SPRING
- Task-specific scenarios
- Overview of network connections
- Development requirements
- Access to SPRING software

## SPRING built in capabilities

- Interactive 3-D graphics world
- 3-D tissue representation + physics
  - Accepts standard data formats
  - Supports deformable models
  - Collision detection and resolution
- Tools with programmed behavior
  - 3-D models of common laparoscopic instruments
  - Behaviors include grasp, probe, cut, etc.

## More things built in

- Configurable via text files:
  - Tissue models, dynamics, graphics
  - Tool definitions + active nodes & edges
- Detects and uses multiprocessor systems for parallel physics computations
- Networked 3-D input and haptics devices
  - Force output on collision
  - Access to other services



## Configuring the basics

- .DESC text file:
  - Identifies tissue & tool object files
    - Defines scaling, color, dynamics
  - Sets up environment:
    - Connect tools to devices (e.g., haptics)
    - Global view: lighting, orientation, geometry
    - Initialize GAME with parameters
  - Invoked by *command line* or *user-menu selection*

```

# Scenario myGAME: probing the liver
dilator0.smf
  dynamics: rigid
  behavior: probing
  ambient: .4 .4 .8 1.
  diffuse: .4 .4 .8 1.
  tipnode: 0
  collisionfaces: 288
Livermodel.obj
  dynamics: deformable
internal
  collisionalgorithm: BoundingSphere
  createsensor: haptic_v1 CALGARY#9999 1.0 0.05 5 0 50
  translation: 5.0, 0.0, 30.0
  gamemode: myGAME <Game Parameters>
  attach: 0 dilator0
  newlight: 0 10 -5 10

```

**Defines objects & tools**

**Defines the environment**

## Simulation vs. platform

- To support a simulation *application*
  - Specific tissues
  - Interactive tools: grasper, dilator, ...
  - Specific activities: dissect, irrigate, ...
  - Pedagogy: user metrics, logging results, learning log, teacher's summaries, etc.

## A real simulator needs:

- A goal for each learning module, and *detection* of it is achieved
- One or more states, with transitions based on user's actions.
- Specialized interactions for task(s)
- Metrics on user's choices, mistakes & correct actions, timing, and performance
- Instructions, hints, & feedback on results

## Pieces of a learning scenario:

- GUI - info & control for user
- Tissue & tools: 3-D models & behavior
- Interactions:
- States and goals:
- Statistics and metrics:
- User feedback:
  - Immediate
  - Post-session
  - To learning supervisor

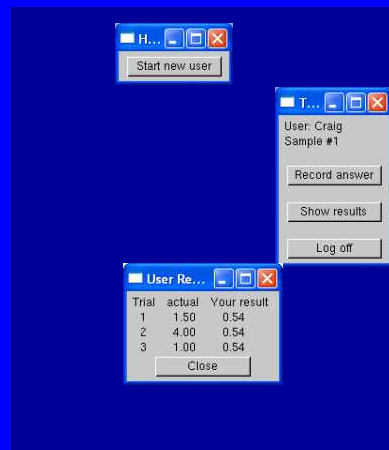
## SPRING Scenario module

- Sets up *scenario* or *game* in SPRING platform.
- Implemented by custom programming using “hooks” into SPRING’s main processing:
  - Initialization
  - Specialized user interface controls
  - 2D & 3D graphics and messages
  - Handling user commands (keyboard)
  - Scenario-specific logic
  - Closing

**Does not change SPRING core code!**

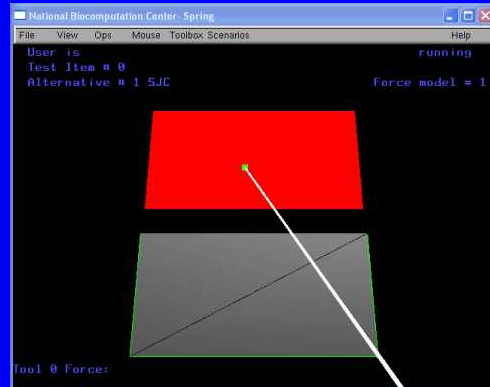
## Adding scenario interface:

```
StartSPRING() {  
    // Initialize basic  
    //  
    SPRINGinterface  
    Scenario->CreateGUI();  
}
```



## Adding scenario graphics:

```
Display() {  
  // Update 3D view  
  - drawObjects();  
  Scenario->Draw2D();  
  Scenario->Draw3D();  
}
```



## Scenario Update: Logic

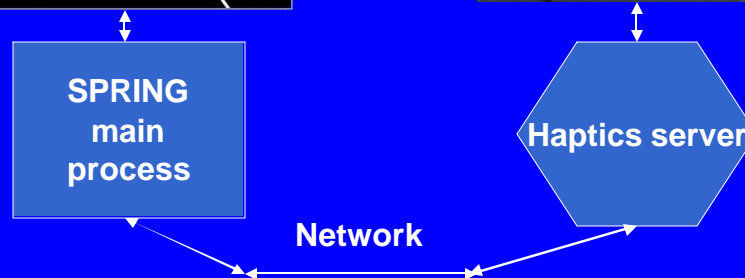
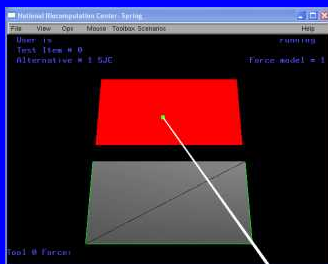
- Detects higher level actions based on collisions
  - E.g., object is “grasped”
- Based on user’s actions, updates metrics of time, performance, errors
- Detects state changes based on
  - interaction of tools and objects
  - Time, performance metrics or other factors

# Connecting scenario logic

```
Update()  
{  
  readSensors();  
  handleCollisions();  
  computePhysics();  
  generateHaptics();  
  
  Scenario->Update();  
}
```

```
myScenario::Update()  
{  
  // Check interactions  
  //   of objects  
  // Update metrics  
  // State changed?  
  // Update state...  
}
```

# Connecting to haptics



## SPRING Open Source

- Sourceforge.net/
  - SPRING Main program
    - Windows, LINUX, UNIX versions (OS X)
  - HapticServer: connects to 3-D input and force output
    - Phantom Desktop, Phantom OMNI (1 or 2)
    - Immersion Laparoscopy Workstation (bimanual)
  - Other servers: display, geometry, ...
- spring.stanford.edu

## Development requirements

- OS: Windows, LINUX, Sun Solaris, (UNIX)
- Development environment:
  - Visual Studio V6, .Net, 2005
  - Make, (Xcode)
- Memory: the more, the better
- CPU:
  - 2GHz good, 4GHz better
  - >1 CPU - allows threads for display and physics

## Conclusions

- Configuration for C++-free development
- Scenarios allow easy development of games
- Painless integration into SPRING platform
- Network services extend connectivity
- It *can* be done!

## Thanks

- This work is partially supported by  
National Library of Medicine Contract  
N01-LM-3-3512, "Advanced Network  
Infrastructure for Distributed Learning  
and Collaborative Research (HAVNet)