Collision Detection and Teamcenter Haptics: CATCH

May 14-30: Logan Scott, Matt Mayer, James Erickson, Paul Uhing, and Tony Alleven

What is a haptic device?

- Haptics
- Delivering haptics in other ways
- Force feedback

Problem Statement

- Manipulate 3D models in a pre-existing model viewer with a haptic input device.
- Collision detection + haptic feedback
- Proof of concept design

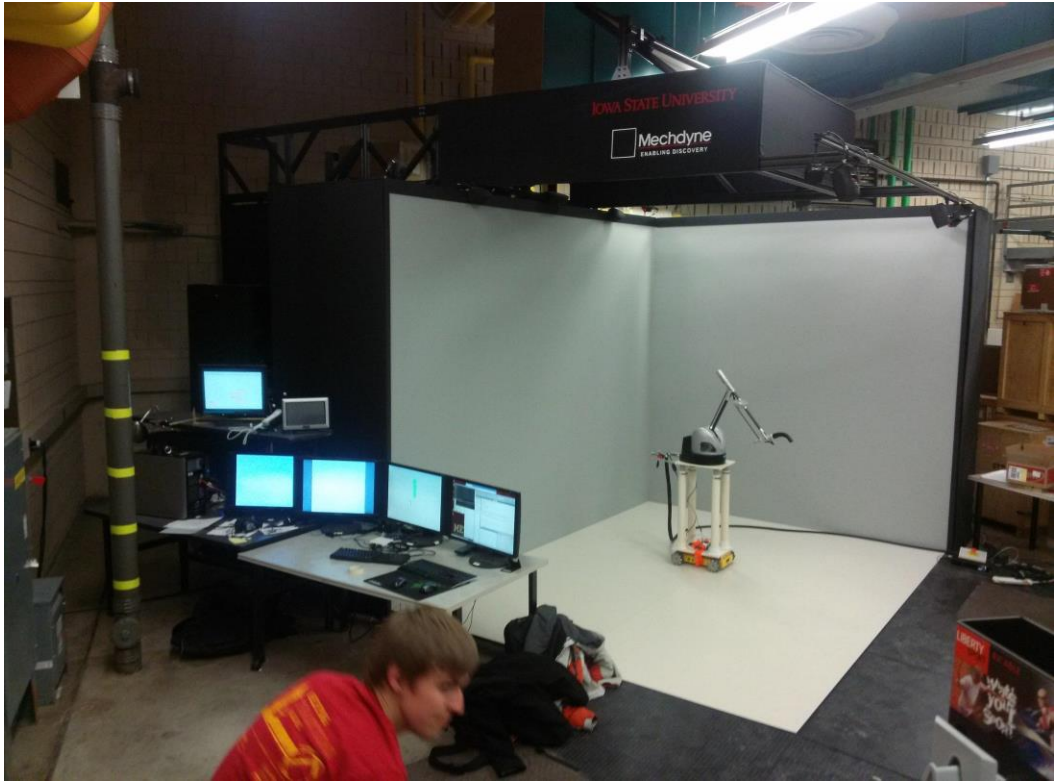
Our Solution

- A C++ library
- Load 3D models into an existing physics engine
- Poll simulation state from the physics engine
- Convert transformation representations
- Update part transformations in the visualizer

Demo

https://www.dropbox.com/s/pbfylg39giai8de/Demo.mp4

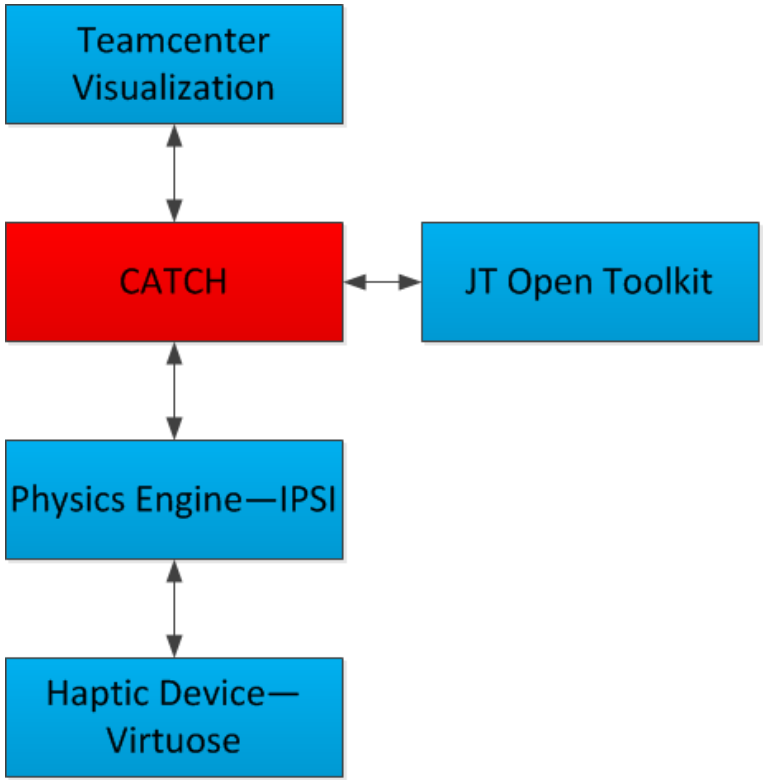
METaL



Dependencies

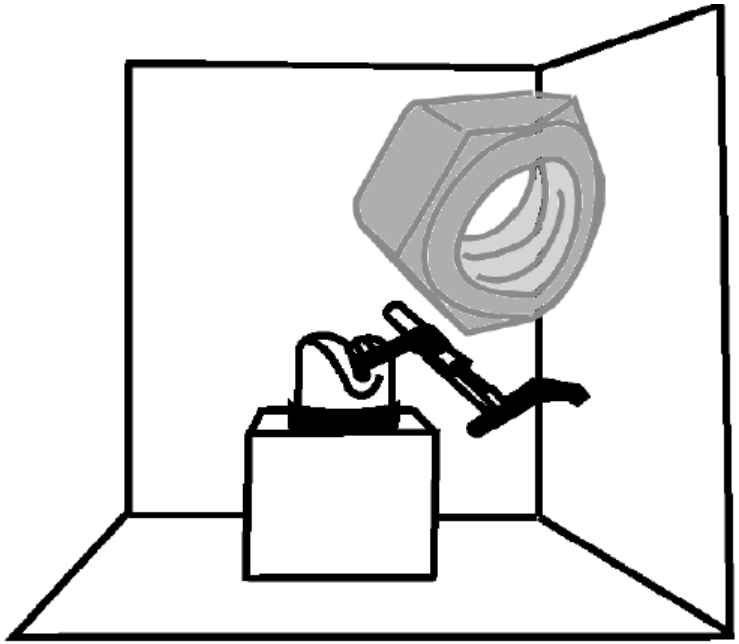
Term	Description	Developed by
CATCH	Our project - what we've built	Us
Teamcenter Visualization (TCVis)	Visualization software	Siemens
VisController	API for interaction with Teamcenter	Siemens
JT	A data format for 3D models	Siemens
JT Open Toolkit	API for interaction with JT files	Siemens
Virtuose	Haptic arm	Haption
IPSI	Physics engine	Haption

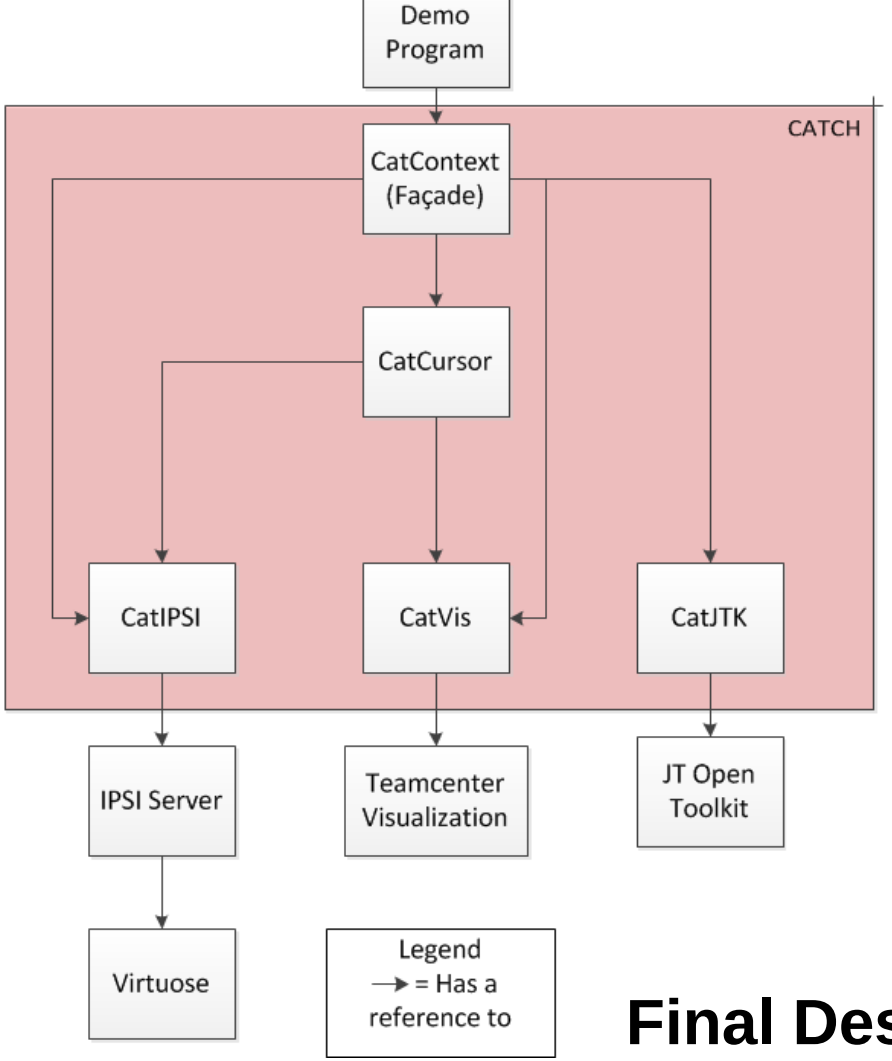
Where does CATCH fit in?



Use Cases

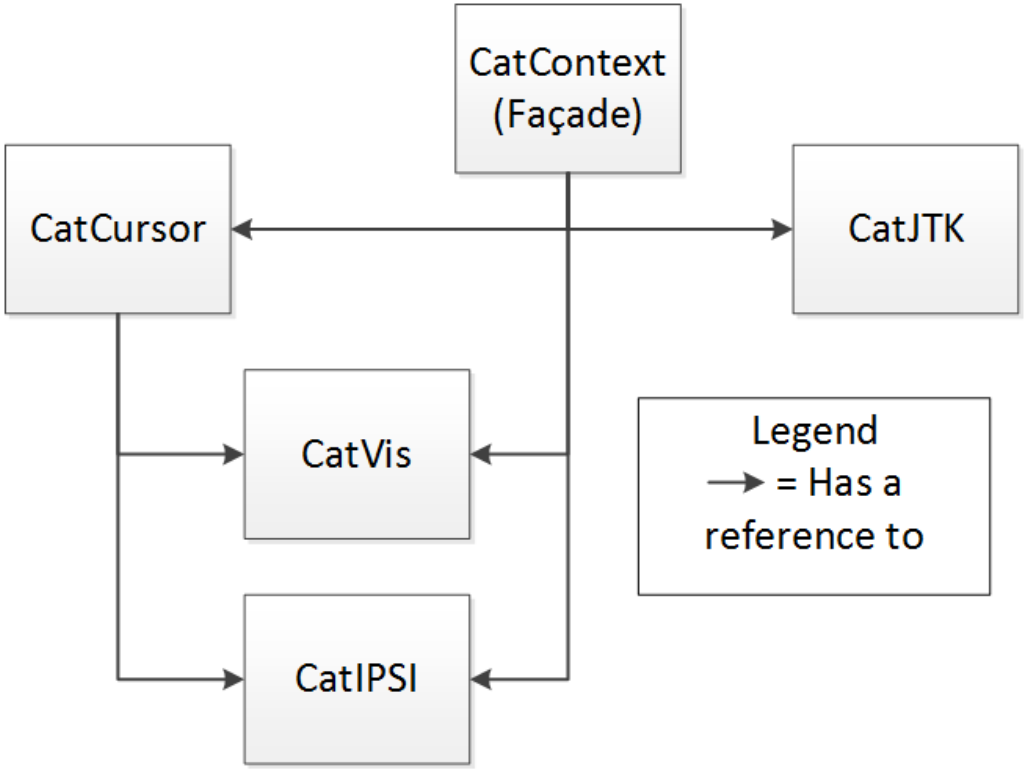
- Internal experimentation
  - Siemens
  - VRAC





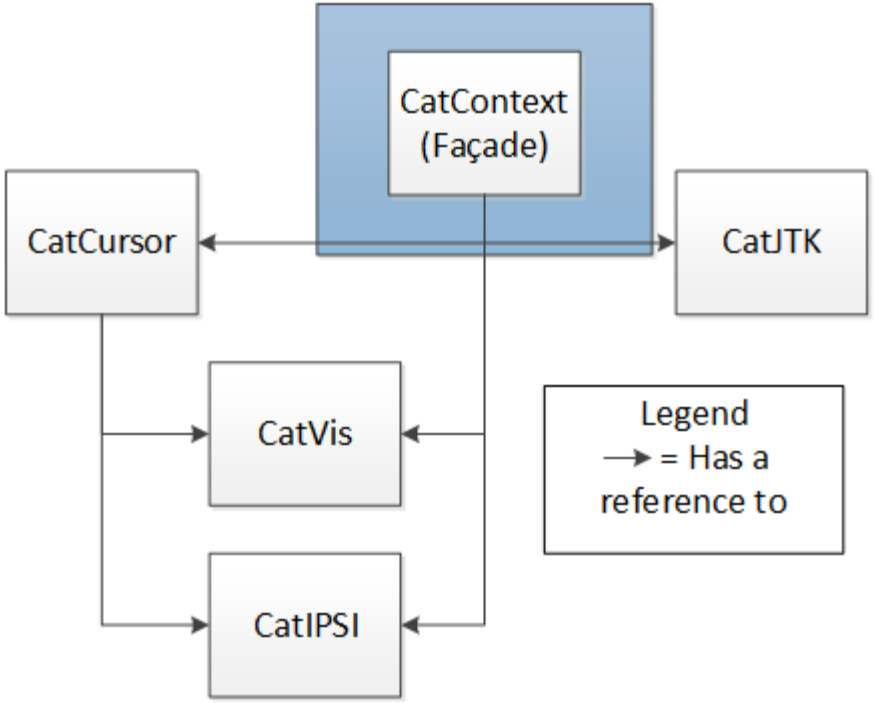
Final Design

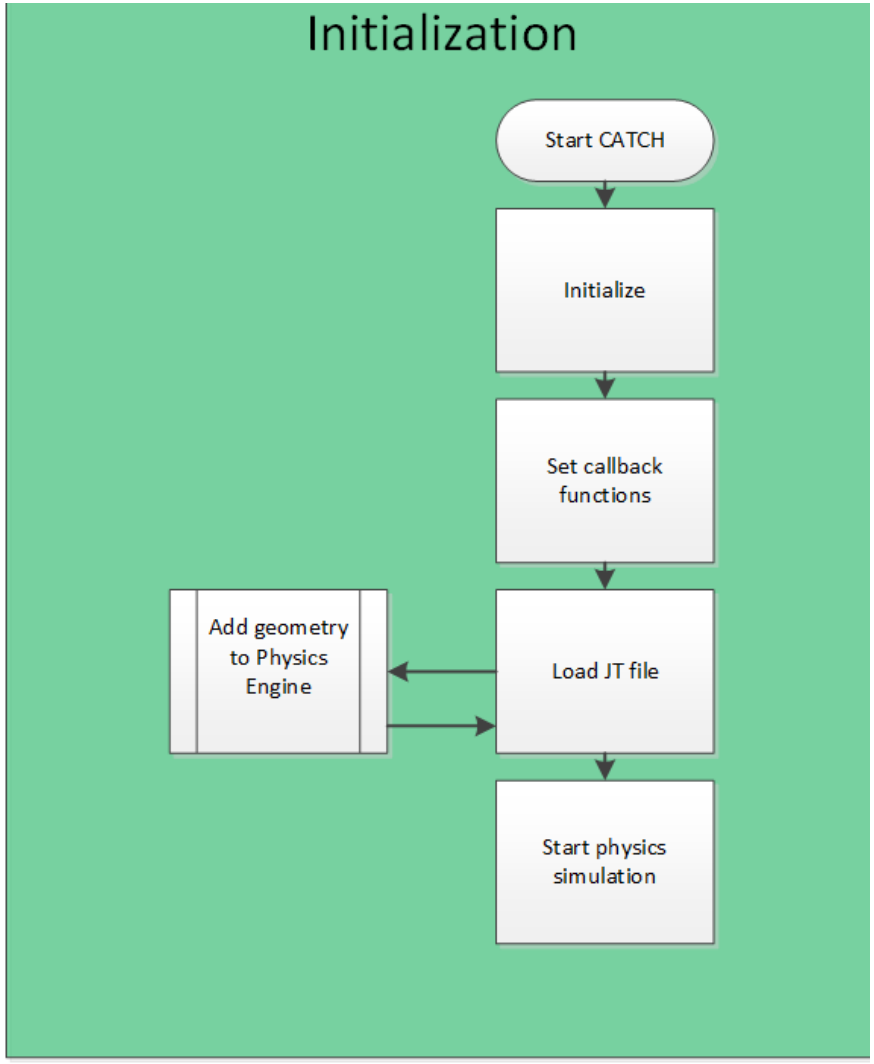
Core CATCH Modules



Facade Module

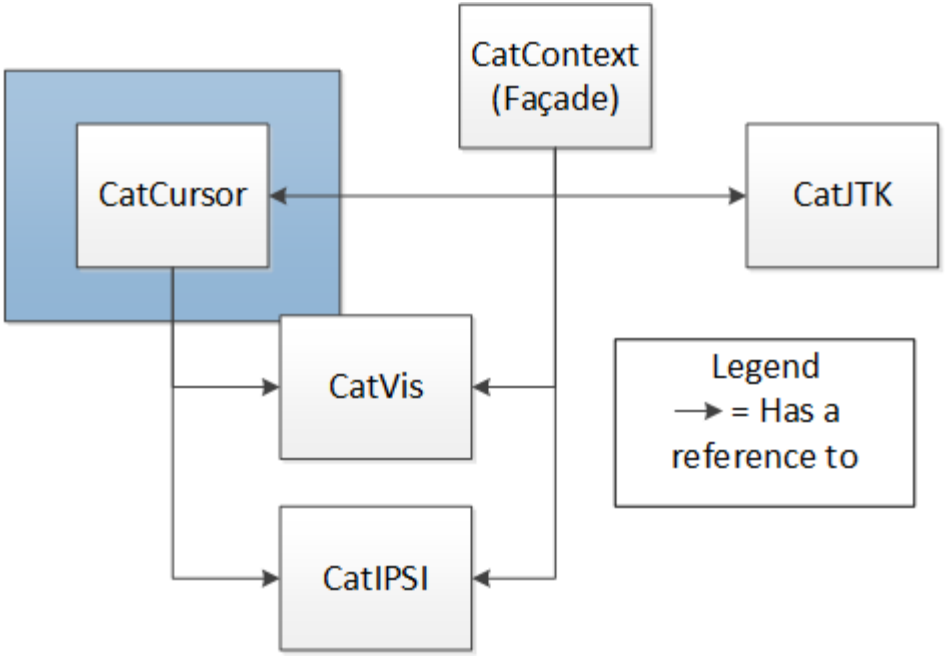
- User-facing
- Inter-module callbacks exist here

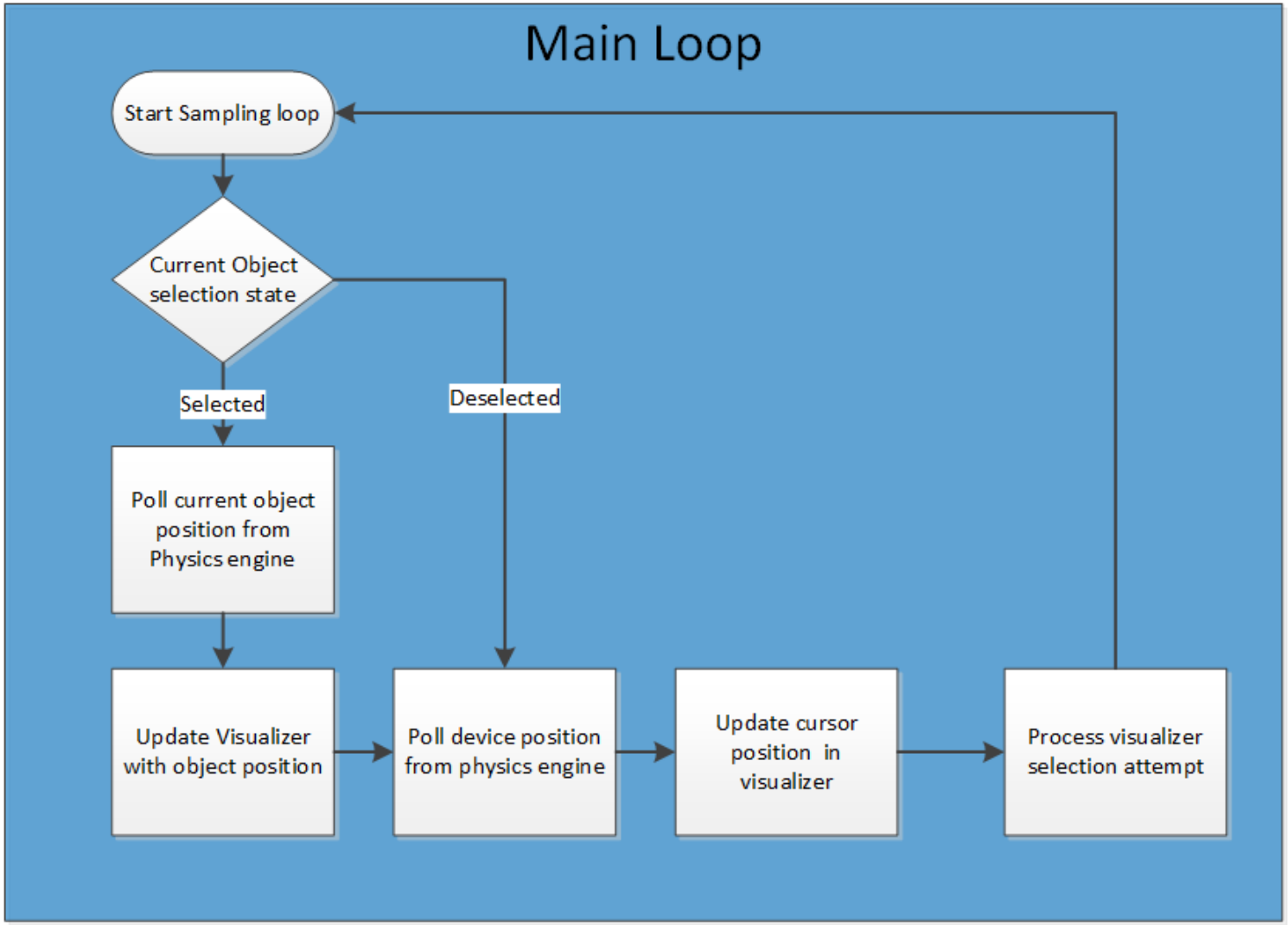


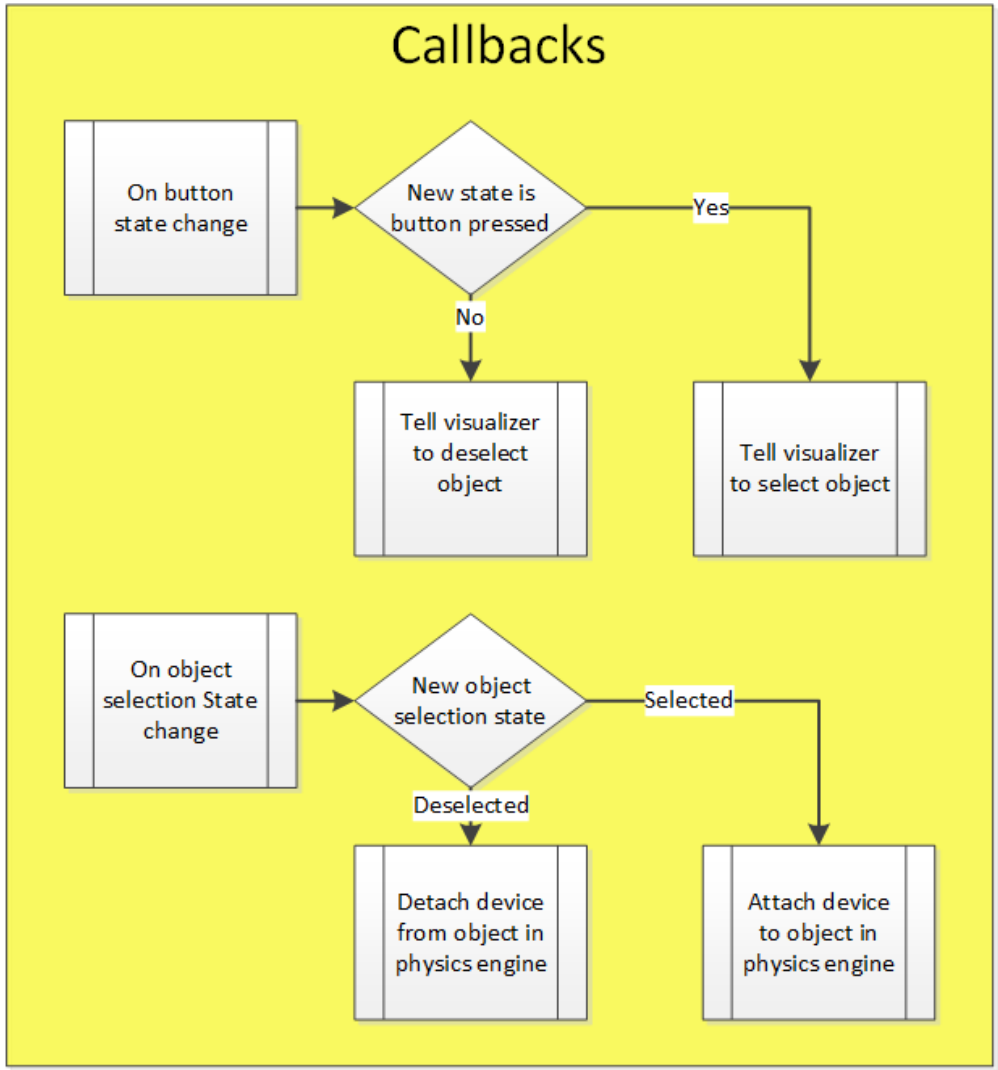

Initialization
Start CATCH
Initialize
Set callback functions
Load JT file
Add geometry to Physics Engine
Start physics simulation

Main Loop Module

- Govern execution speed
- Transport cursor / part transformations

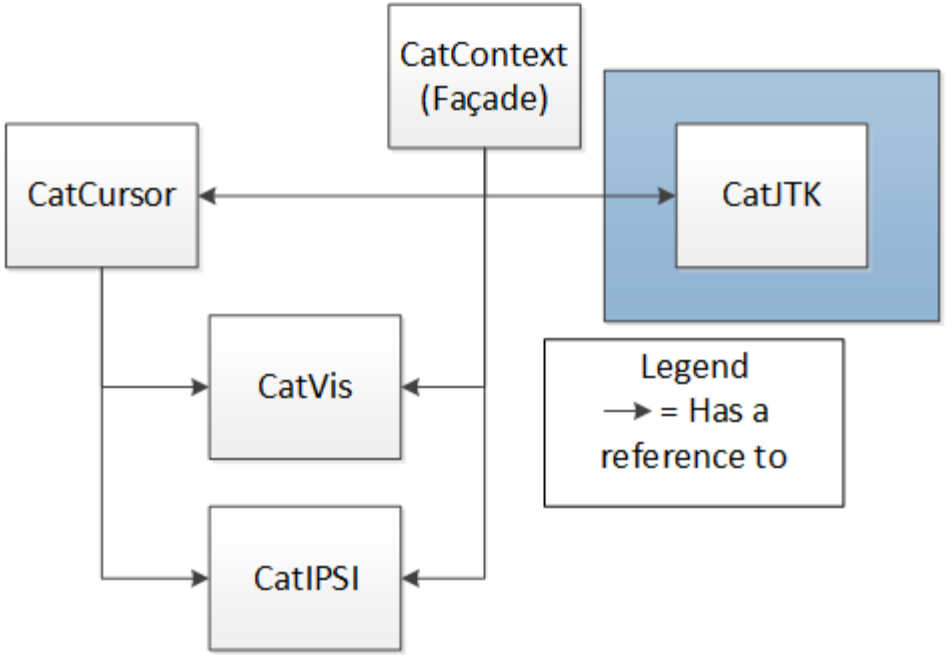



Main Loop
Start Sampling loop
Current Object selection state
Selected
Deselected
Poll current object position from Physics engine
Update Visualizer with object position
Poll device position from physics engine
Update cursor position in visualizer
Process visualizer selection attempt


Callbacks
On button state change
New state is button pressed
Yes
No
Tell visualizer to deselect object
Tell visualizer to select object
On object selection State change
New object selection state
Selected
Deselected
Detach device from object in physics engine
Attach device to object in physics engine

JT Open Toolkit Module

- Generate 3D part data
  - Triangle meshes
  - Transformations
  - IDs



Transformation Representations

Representing a transformation as a double[16]

OpenJTTK and VisController (File Importer and Visualizer API)

3x3 Rotation

0	1	2	12
3	4	5	13
6	7	8	14
9	10	11	15

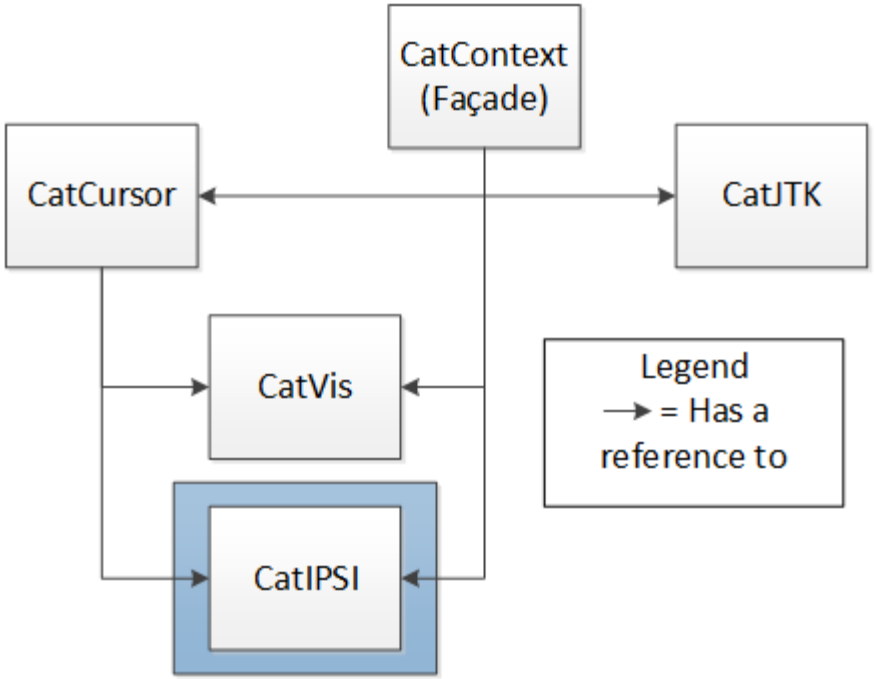
X,Y,Z Translation

IPSI (Physics Engine) representation

0	3	6	12
1	4	7	13
2	5	8	14
9	10	11	15

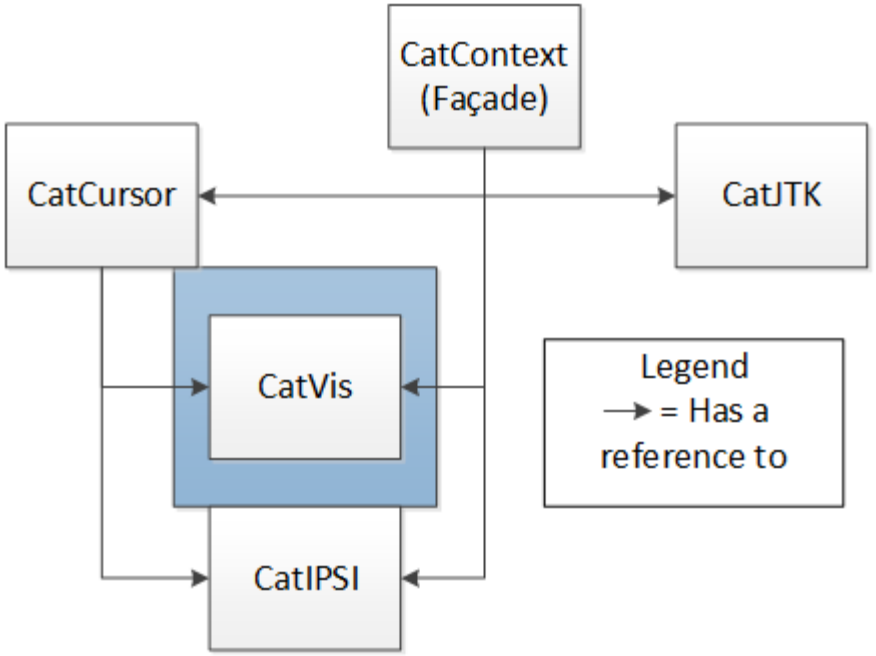
IPSI (Physics) Module

- Update and sync with physics engine
  - 3D parts
  - Haptic device



Visualization Module

- Update and sync visualizer
- Communicate through VisController API



Development Process

- Heavy design work at the beginning
  - Paid off at the end
- Iterative process
  - Weekly tests
  - Space mouse
- Strived for usable demos

Results

- CATCH allows physical feedback in a virtual environment
- Some limitations
  - IPSI isn’t designed for tightly-packed assemblies.
  - Licensing issues (10 or more help tickets)
- Able to affect changes in VisController

Special Thanks

Dr. Tsung-Pin Yeh

Dr. Judy Vance

Dr. David Weiss

Jerome @ Haption

Questions?

Functional Requirements

Functional requirements

1. Manipulate a cursor in Teamcenter Visualization with a haptic device
2. Select an object in Teamcenter Visualization with the haptic device

Functional Requirements

Functional requirements(cont.)

3. Object must have appropriate haptic feedback upon a collision with another object
4. Support loading part geometry via Jupiter Tessellation (JT) files

Non Functional Requirements

Non-functional Requirements

1. I/O Lag-time <200 ms
2. After accounting for lag time, all object models shall be synchronized.
3. Document to an extent that others can use the library

Use Enviroment

- Proof of concept
  - Windows 7(x64) only
  - Simple 3D assemblies
  - Virtuose haptic device
  - METaL

Smartsheet Link

https://app.smartsheet.com/b/publish?EQBCT=6fad3999ac99408494af29cc2b90f239

Technical Challenges

Knowledge Acquisition

- IPSI
- VisController
- Virtuose Availability

Technical Challenges

Physics Engine Representation ≠ Visual Representation

- Differences in Base Frame
  - Relative vs. Absolute
- Memory Representation
- Solution: Physics Engine ⇔ Visualization: Consistency in Transformation Process

Technical Challenges

Data Ownership and Lifecycle

- First major C++ application
- Memory management
- Access from concurrent threads
  - Heap-heavy allocation: guarantees memory remains in scope

High-Level Overview

- Goal: Create prototype application that uses a haptic controller arm (Virtuose) to manipulate parts within a 3D scene
  - Rendered in TeamCenter Visualization
  - Physics and haptic feedback performed by IPSI
- Proof of Concept: Commercial Software can be used in a haptic application.

Scope

- Integrating the haptic arm, the physics engine, and displaying the virtual scene
- This is proof of concept
- We are not making a plug and play haptic device/CAD viewer integration application

Testing

- Prototyping
  - Verification: Space Mouse and Virtuose Simulator
  - Validation: Two-week sprints + client demo
- METaL Lab
  - Virtuose: In France for repairs
  - Upon return, test with physical hardware

Rewritten to match the form of a classical 4x4 array

0	1	2	12
3	4	5	13
6	7	8	14
9	10	11	15

Indices [12], [13], [14] are the x, y, and z translation

Indices [0] – [8] represent a row major 3x3 rotation matrix

Indices [9], [10], [11] are typically 0, and [15] is typically 1

Rewritten to match the form of a classical 4x4 array

0	3	6	12
1	4	7	13
2	5	8	14
9	10	11	15

Indices [12], [13], [14] are the x, y, and z translation

Indices [0] – [8] represent a column major 3x3 rotation matrix

Indices [9], [10], [11] are typically 0, and [15] is typically 1

VisController, OpenJTToolkit, and IPSI
Quaternion Representation

As double[4]
Stored as scalar last

Q = a + bi + cj + dk

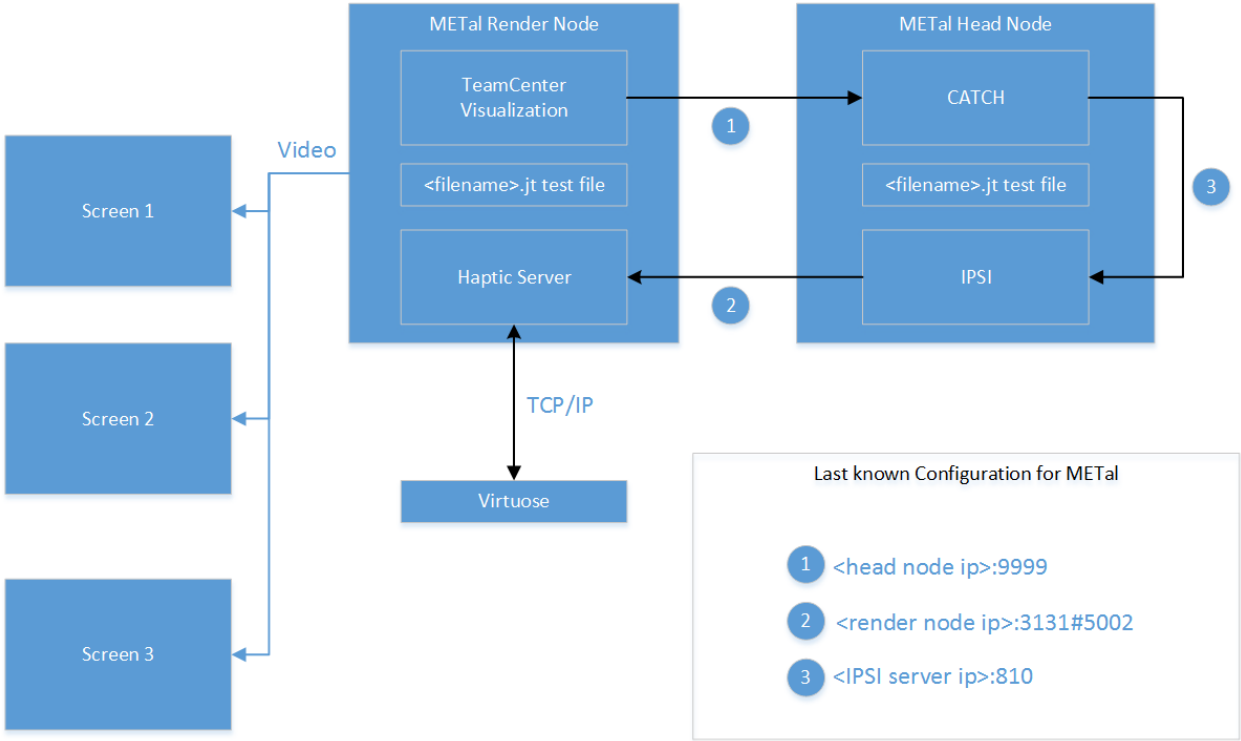
Rewritten to match the form of

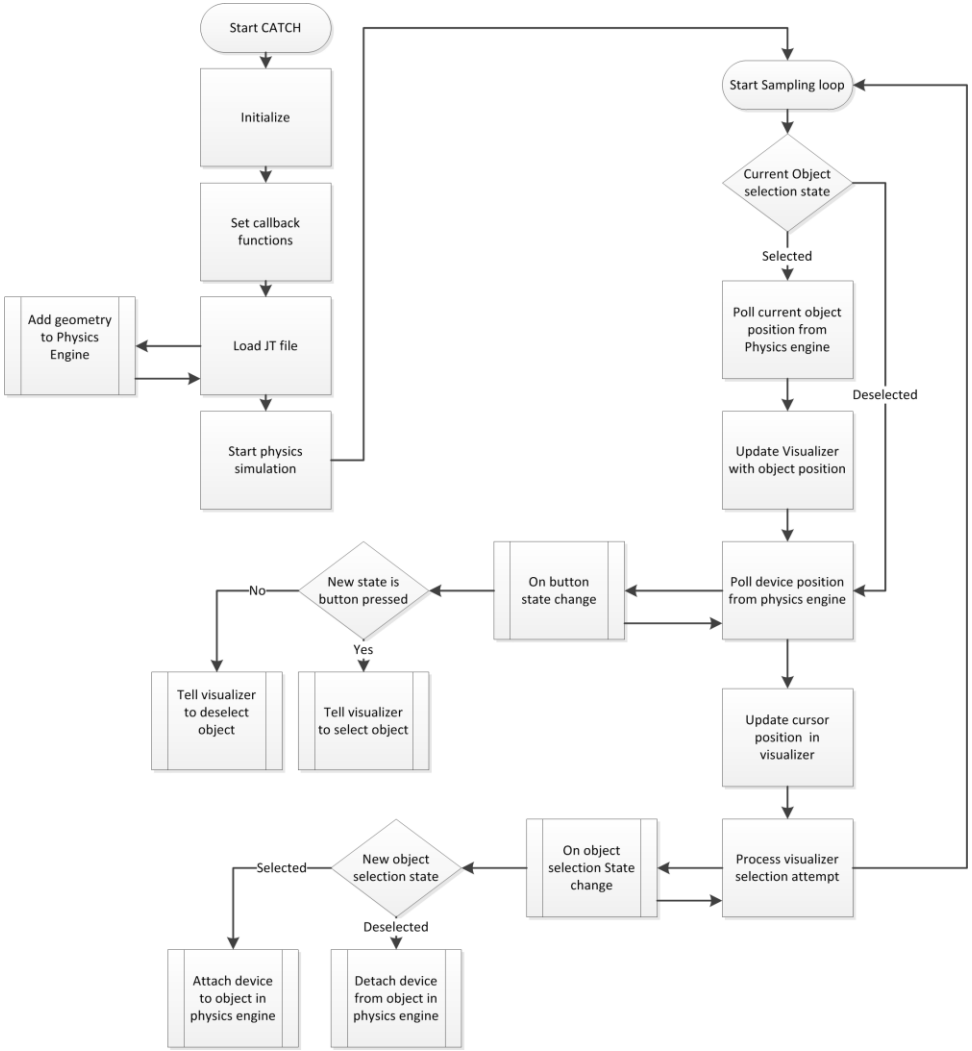
IPSI double[7] Transformation Representation #2

Array is split into Position and Orientation

double[7] M
Position:
M[0] = x coord
M[1] = y coord

METal layout




Start CATCH
Initialize
Set callback functions
Load JT file
Add geometry to Physics Engine
Start physics simulation
Start Sampling loop
Current Object selection state
Selected
Deselected
Poll current object position from Physics engine
Update Visualizer with object position
Poll device position from physics engine
On button state change
New state is button pressed
No
Yes
Tell visualizer to deselect object
Tell visualizer to select object
Update cursor position in visualizer
Process visualizer selection attempt
On object selection State change
New object selection state
Selected
Deselected
Attach device to object in physics engine
Detach device from object in physics engine