Table of Contents

Table of Contents ii
Revision History iv
1. Introduction 1
1.1 Purpose 1
1.2 Intended Audience and Reading Suggestions 1
1.3 Project Scope 1
2. Overall Description 1
2.1 Product Perspective 1
2.2 Product Features 1
2.3 Operating Environment 1
2.4 Design and Implementation Constraints 1
2.5 User Documentation 1
2.6 Assumptions and Dependencies 2
3. Interface Requirements 2
3.1.1 IGSTK components shall be limited to 8 exposed methods. 2
3.1.2 Interactions with IGSTK components that require more than request/response protocol must be explicitly documented in the component API specification. 2
4. Functional Requirements 2
4.1 Development Environment-Specific 2
4.1.1 IGSTK components shall execute a high-level task autonomously requiring minimal interaction. 2
4.1.2 IGSTK components shall not expose underlying class structure in header files. 2
4.2 Application-Specific 2
4.2.1 At a minimum, IGSTK shall support general components for tracker capabilities, viewer capabilities, segmentation, and registration. 2
4.2.2 The IGSTK platform shall include a minimum of two application examples. 2
5. Infrastructure and Nonfunctional Requirements 2
5.1 Performance Requirements 2
5.1.1 IGSTK components that manage events shall include threading capability. 2
5.2 Test and Evaluation Requirements 2
5.2.1 A test case shall be included for each IGSTK component. 2
5.2.2 Each IGSTK components must have 100% testing coverage (explain this). 2
5.3 Implementation and Transition Requirements 2
5.3.1 IGSTK components shall enable compilation via CMake. 2
5.3.2 Each IGSTK components shall be documented with corresponding DOxygen files 2
5.4 Quality Assurance and Reliability Requirements 2
5.4.1 Each IGSTK component must be controlled by the IGSTK internal state machine code. 2
5.5 Configuration Management Requirements 2
5.5.1 Each configuration management action must be supported by a requirement number. 2
5.6 Communication Requirements 2
5.6.1 The IGSTK platform shall contain a general communication component that acts as a proxy to all communication ports to and from the application such as the serial/parallel ports or wireless connections. 2
5.7 Logging Requirements 2
5.7.1 The IGSTK platform shall contain a general, centralized logging capability that can be used for all IGSTK components. 2
5.7.2 The logging capability shall support timestamps for the audited information. 2
5.7.3 The logging capability shall enable logging for multiple components concurrently and support auditing into multiple files. 3
5.8 Timing and Latency Requirements 3
5.8.1 Add synchronization requirements. 3

5.9 Error-Handling and Notification Requirements 3
5.9.1 Add error-handling and notification requirements. 3
6. Tracker Requirements 3
6.1 Functional Requirements 3
6.1.1 The tracker component shall have the ability to initialize the tracker hardware. 3
6.1.2 The tracker component shall have the ability to reset the tracker hardware 3
6.1.3 The tracker component shall have the ability to start and stop tracking independent of the tracker hardware. 3
6.1.4 The tracker component shall have the ability to restart the tracker hardware. 3
6.1.5 The tracker shall incorporate a state machine that, at a minimum, supports the following states: IdleState, CommunicatingState, ToolsActiveState, TrackingState, SetUpCommunication, SetUpTools, StartTracking, UpdateToolsStatus, StopTrackingCommunicatingState 3
6.2 Nonfunctional requirements 3
6.2.1 The tracker component shall have an integrated module to support connectivity to the logging capability. 3
6.2.2 The tracker component shall have an integrated module to support the loading configuration files and the management of the configuration data. 4
6.2.3 The integrated configuration file management component must support configuration files in the Extensible Markup Language (XML). 4
6.2.4 The tracker component shall have a non-limiting design to support at least 4 tools connected to a specific tracker hardware. 4
6.2.5 The tracker component will verify if the rate requested by the user is lower or equal than the rate supported by the tracker, if the requested rate is higher then the tracker component should create an exception. 4
6.2.6 The tracker component shall support diagnostic procedures that occur at the frequency designated by the application developer to determine when a tool is substituted or disabled. 4
6.2.7 The tracker component shall have multiple levels of diagnostic support. At the highest level, a component and human diagnostic must be executed at configuration time. 4
6.2.8 The tracker component shall have an open interface to the viewing component using a universally-acceptable data structure. (New position class) 4
6.2.9 The tracker component interface should send messages at a minimum comprising position, orientation, quality (error), number of tools, and tool identification. 4
6.2.10 The tracker component shall report the maximum refresh rate and time latency on demand. 4
6.2.11 The tracker component shall support a hot swapping routine that consists of …. All tracker tools must be installed and verified at initialization time. Tracker will only support hot swapping for tools that enable this capability. 4
6.2.12 The tracker component must confirm that the tools are at the same refresh rate as the tracker hardware. 4
7. Viewer Requirements 5
7.1 Functional Requirements 5
7.1.1 The viewer component will have the capability to visualize a tree structure representing a scene. 5
7.1.2 The viewer component will, at a minimum, operate with the tree structure implemented within ITK Spacial Object and abstracted by IGSTK Representation classes. 5
7.1.3 The viewer component should allow the content of the scene to be encapsulated within the hierarchy of the IGSTK Representation classes. 5
7.1.4 The viewer shall provide slice selection, panning, zooming, and rotation as associated with the functionality of the camera. 5
7.2 Nonfunctional Requirements 5
7.2.1 Each viewer shall be capable of rendering , at a minimum, axial, saggital, coronal, and three-dimensional views. 5
7.2.2 Multiple separate viewers should not limit the ability to be composed into a context-specific application framework. 5

7.2.3 The viewer is a passive class that is unaware of the internal scenes captured in the representation structures. 5
8. Scene Representation Requirements 5
8.1 Functional Requirements 5
8.1.1 The IGSTK platform should contain a representation structure that, at a minimum, captures a parallel representation of a hierarchy ITK Spacial objects. 5
8.2 Nonfunctional Requirements 6
8.2.1 The representation structure should not have specific hard connections to the viewer component. 6
8.2.2 The representation structure should be able to polymorphically handle its underlying scene objects. 6

Revision History

Name	Date	Reason For Changes	Version
Blake, Brian	8/202004	Initial Document Creation	1.1
Blake, Brian	10/28/2004	First CVS Check-In	