|  |
| --- |
|  |
| Tunnel-K Software Design Description (SDD) |
|  |
|  |
| **Josh Calahan, Wes Cothran, Chris Davis, Michael Lynch, Brian Pittman** |
| **12/1/2011** |

**Approvals**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_**

Instructor Approval: Dr. Jeffrey Kulick Date

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_**

Team Member: Josh Calahan Date

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_**

Team Member: Wes Cothran Date

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_**

Team Member: Chris Davis Date

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_**

Team Member: Michael Lynch Date

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_**

Team Member: Brian Pittman Date

**Revisions**

|  |  |  |
| --- | --- | --- |
| **Revision Number** | **Date** | **Description** |
| Initial | 12/01/11 | Original Draft |

Contents

1. Scope 5

1.1. Identification 5

1.2. System overview 5

1.3. Document overview 5

2. Referenced documents 5

3. System item-wide design decision 6

4. Software item architectural design 6

4.1. Software item components 6

4.2. Concept of execution 6

4.3. Interface design 7

4.3.1. Interface identification and diagrams 7

4.3.2. Project-unique identifier of interface 7

5. Software item detailed design 9

5.1. (Project-unique identifier of a software unit, or designator of a group of software units) 9

6. Requirements traceability 10

7. Notes 10

8. Annexes 10

### Scope

### Identification

Tunnel-K is an effort by graduate software engineering students at the University of Alabama in Huntsville enrolled in a two-semester Software Engineering Studio course (CPE656 and CPE658). The endeavor is aimed at building a small-scale wind tunnel and associated software systems and is intended to be used by science museums, schools, etc. for educational purposes. The work is being done in association with the Hands-On Science Center (HOSC) in Tullahoma, TN, and the team will draw upon their professional experience working as professional software engineers.

### System overview

The overall Tunnel-K system consists of a wind tunnel structure along with associated computer hardware and software, wiring, sensors, motors, fans, power supplies, etc. used for controlling and monitoring the operation of the tunnel. Additionally, a two-dimensional flow solver application suite will provide the opportunity for experimentation with various shapes in a virtual wind tunnel environment and graphically displayed mach and pressure gradients. The suite will also provide integration with the physical wind tunnel controls so that simulated conditions and be illustrated in the real world.

### Document overview

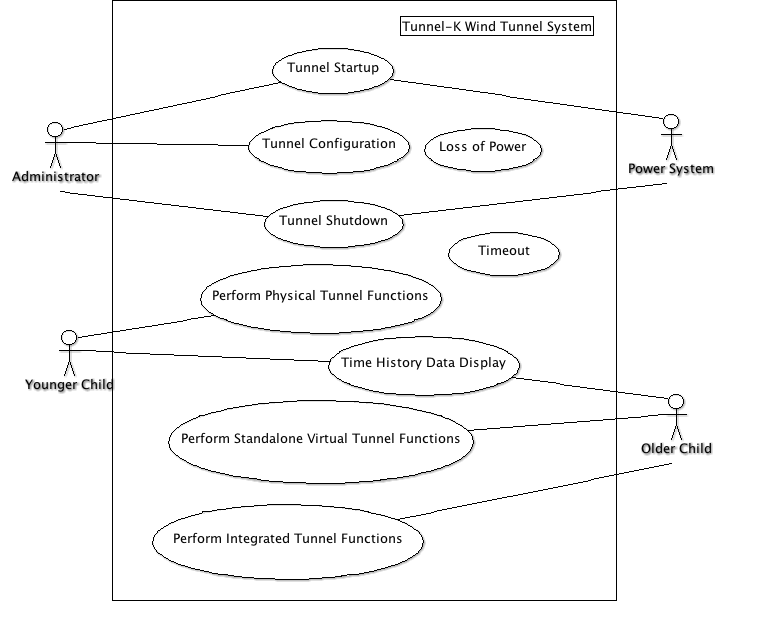
This document addresses system design issues bridging the gap between requirements and implementation. Elements in this document will tie individual requirements to their respective design issues drawing heavily on the Unified Modeling Language and existing prototypes.

### Referenced documents

* Tunnel-K Statement of Work (SOW)
* Tunnel-K Rough Order of Magnitude Estimate (ROM)
* Tunnel-K Software Requirements Specification (SRS)
* Tunnel-K Software Development (SDP)
* Tunnel-K Google Code site, <http://code.google.com/p/tunnelk>
* Tunnel-K Google Groups site, <http://groups.google.com/group/tunnelk>
* Trial-Use Standard for Information Technology Software Life Cycle Processes Software Development Acquirer-Supplier Agreement, J-STD-016-1995
* UAH CPE656 Fall 2011 Course Syllabus, Dr. Jeffry Kulick
* Code Conventions for the Java Programming Language, <http://www.oracle.com/technetwork/java/codeconvtoc-136057.html>
* Code Style Guidelines for Contributors, (Google Android Style Guide) <http://source.android.com/source/code-style.html>
* Writing a Library for Arduino, <http://arduino.cc/en/Hacking/LibraryTutorial>

### System item-wide design decision

Foremost, the team will use engineering judgment will make design decisions. UML will be used to model the system in order to refine the design on paper prior to making implementation level decisions. An appropriate balance between the hardware wind tunnel and the software wind tunnel will be sought. Here is a use case diagram of the complete system:



For a complete description of the actors and use case details, please see the system requirements document.

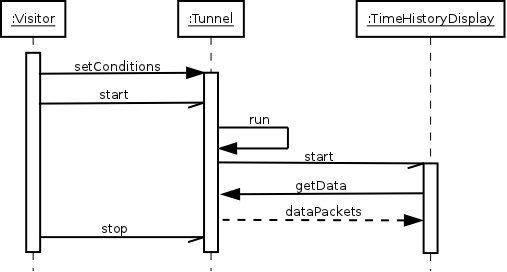
### Software item architectural design

### Software item components

* + 1. Virtual Wind Tunnel
    2. Application Controller
    3. Arduino Routines

### Concept of execution

The following sequence diagram shows interaction between a visitor (actor), a tunnel (physical or virtual), and a time history display). The visitor applies the desired tunnel conditions then indicates that the tunnel is to begin operation. The tunnel responds by “running” then indicating to the history plots to begin displaying. The history plots query data from the tunnel and display time history plots to the user.

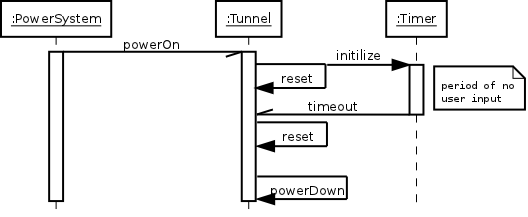
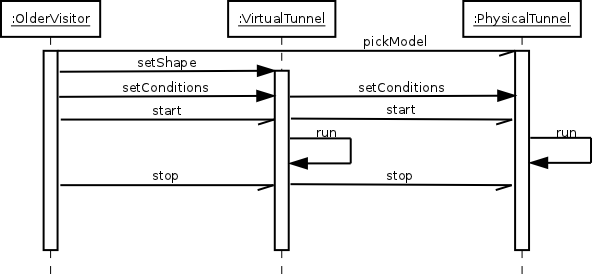


### Interface design

### Interface identification and diagrams

These diagrams focus on the workflow of the Tunnel-K system as an exhibit. Each diagram is marked with a design designator use for requirements traceability.

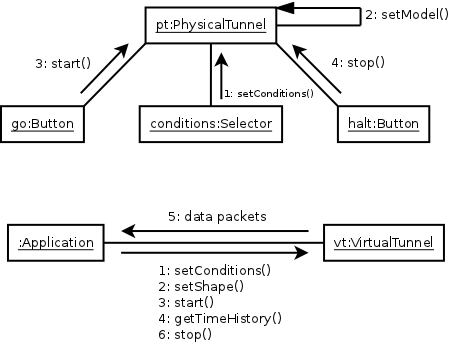
### Project-unique identifier of interface

* + - 1. Power system and timer interface
      2. Integrated physical/virtual interface
      3. Analysis of alternatives for considering software vs. hardware controls.

These design elements were developed to visualize implications of supplying hardware controls in addition to pure software controls for the Tunnel-K system.

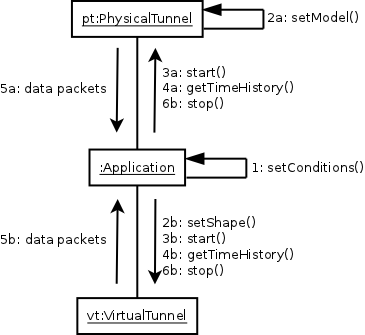
* + - * 1. Option # 1 - No link between physical and virtual solver

This diagram shows an unlinked chasm between the physical and virtual tunnels. The first sub-model shows physical buttons and selectors that operate on the physical tunnel. The second sub-model is a simplification of the first model, an application controlling only a virtual tunnel.



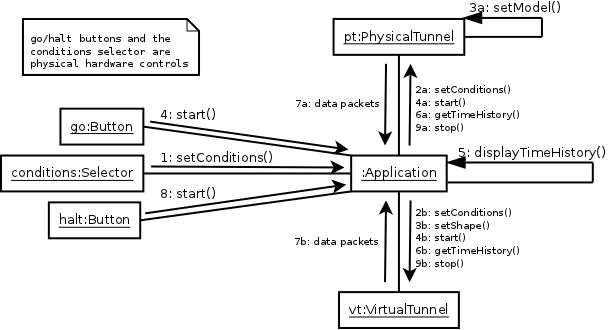
* + - * 1. Option # 2 - Only software controls

This diagram shows the pure software controls option. A single application (perhaps a mobile device application) controls both the physical and virtual tunnels. The first step is applying conditions to the tunnels. Step two is selecting a model for the case of the physical wind tunnel and setting a shape for the virtual one. Step three tells to tunnels to go. Step four and five displays time history. Finally, step six terminates tunnel operation.



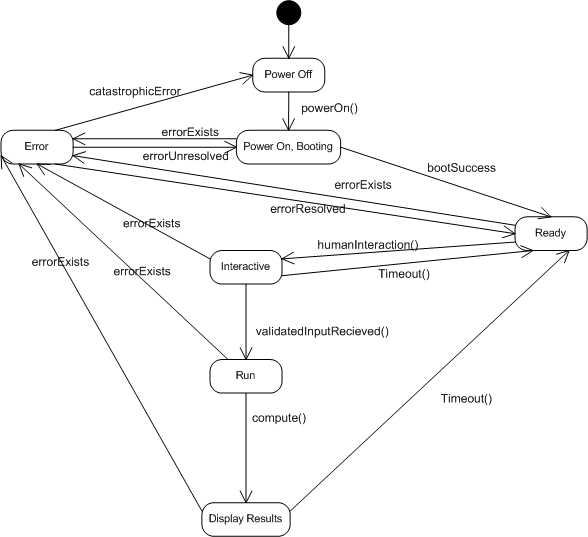
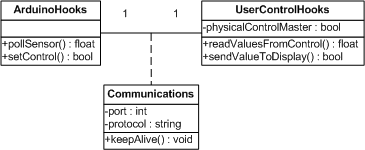
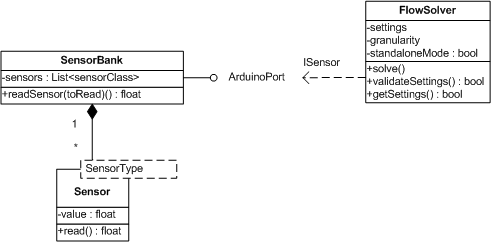
* + - * 1. Option # 3 - Software and hardware controls

This third communication diagram combines the first two communication diagrams and is a first-step proposal to the Tunnel-K stakeholders on how hardware controls and pure software might co-exist.



### Software item detailed design

### (Project-unique identifier of a software unit, or designator of a group of software units)

* + 1. Detailed state diagram addressing power, interaction, operation, and display. 
    2. Arduino relationship with configuration level plug-in interface. 
    3. Communication interface between the Arduino tunnel system and the controls and display tied to the tunnel 

### Requirements traceability

This table maps requirements listed in section 3 requirements document with design solutions listed in the section 4 and 5 of this document.

|  |  |
| --- | --- |
| Requirement(s) | Addressed by Design Elements |
| 3.2.1. | 4.2. |
| 3.2.2. | 4.3.2.3. |
| 3.2.3. | 4.3.2.3. |
| 3.2.4. | 4.3.2.2. |
| 3.2.5. | 4.2. |
| 3.2.6. | Addressed during implementation |
| 3.2.7. | Addressed during implementation |
| 3.2.8. | 4.3.2.3.1., 4.3.2.3.2., 4.3.2.3.3. |
| 3.2.9. | Addressed during implementation |
| 3.2.10. | 4.3.2.1. |
| 3.2.11. | 5.1.1. |
| 3.2.12. | Addressed during implementation |
| 3.10.1. | 4.3.2.2. |
| 3.10.2. | Addressed during implementation |
| 3.10.3. | Addressed during implementation |
| 3.12.1. | Addressed during implementation |
| 3.13.1. | Addressed during implementation |
| 3.14.1. | Addressed during implementation |
| 3.15.1. | Addressed during implementation |
| 3.16.1. | 4.3.2.1. |
| 3.18.1. | 4.3.2.1. |
| 3.20.1. | Addressed with project documents addressed in section 2 of this document. |

### Notes

None

### Annexes

None