

Design and Implementation  
of a  
Control Protocol  
for a  
Wireless System

K. Monique Bonneville  
(monique@acm.org)

7 August 1998

**Abstract**

The Intercom Protocol project at the University of California at Berkeley is designing and implementing a simple multinode wireless intercom system. This system will support full-duplex voice communication for multiple users. Protocols designed for Intercom should support n-way channels, as well as provide for robust operation in the event of failures. This paper describes the process undertaken in order to design and implement a control protocol that meets the requirements for this wireless system. A detailed explanation for the purpose of this research is given, as well as a review of the overall system. All steps involved in the design of a control protocol from its highest level of design flow to implementation are noted. The feasibility of the chosen language and tools are thoroughly investigated. From this preliminary investigation, it is evident that this particular method has much to be desired. However, a control protocol has been produced that shall be tested within the near future.

## Contents

<b>1 Introduction</b>	<b>4</b>
1.1 Group Background. ....	4
1.2 System Overview. ....	4
<b>2 Control Protocol</b>	<b>6</b>
2.1 Methodology. ....	6
2.2 Research. ....	7
2.3 Background/ Specifications. ....	7
<b>3 Design</b>	<b>8</b>
3.1 Purpose. ....	8
3.2 Modeling. ....	8
<b>4 Implementation</b>	<b>12</b>
4.1 Analyzing. ....	12
4.2 Generating. ....	12
<b>5 Design Flow Feasibility</b>	<b>13</b>
5.1 SDL. ....	13
5.2 Telelogic Tau. ....	13
<b>6 Goals</b>	<b>14</b>
<b>7 Conclusions</b>	<b>15</b>
<b>8 Acknowledgments</b>	<b>16</b>

**List of Figures**

1	Intercom Project. . . . .	5
2	Protocol Stack . . . . .	6
3	Abstract Model. . . . .	9
4	Example of complete system model . . . . .	11

## **1 Introduction**

This paper describes the process used to create a control protocol. It defines the methodology chosen and establishes reasoning for the existence of such research. A subpurpose of this effort was an investigation into the feasibility of a specific design flow. The information contained in this paper is relevant only to this project, and to the research that was conducted by the author.

### **1.1 Group Background**

The main focus of the System Design group at the University of California at Berkeley is to develop an architecture to aid in complex system design. The design stage from conception to completion is being evaluated and tracked in order to provide a smooth transfer of information amongst various designers using various tools and design flows. The design flow, for this research, includes methodologies, software tools, and languages required to complete the design stage.

At least initially, system design concepts will be applied to low power wireless design. The Intercom project (a subgroup of System Design) is focusing on this research. The ultimate goal of Intercom is the development of a voice-based intercom system implemented on a single chip. The short-term goal is to develop communications protocols for the target system on a flexible development platform, also built by Intercom using off-the-shelf components.

### **1.2 System Overview**

The Intercom project is divided into two subgroups: Hardware and Protocols, with two major overlapping areas (refer to Figure 1). The immediate goal of the Intercom Protocol group is

to design and implement a set of protocols for the Intercom system. These protocols will be used as a driver for development of a concept to physical level design environment tailored toward communication protocols for wireless systems in general.

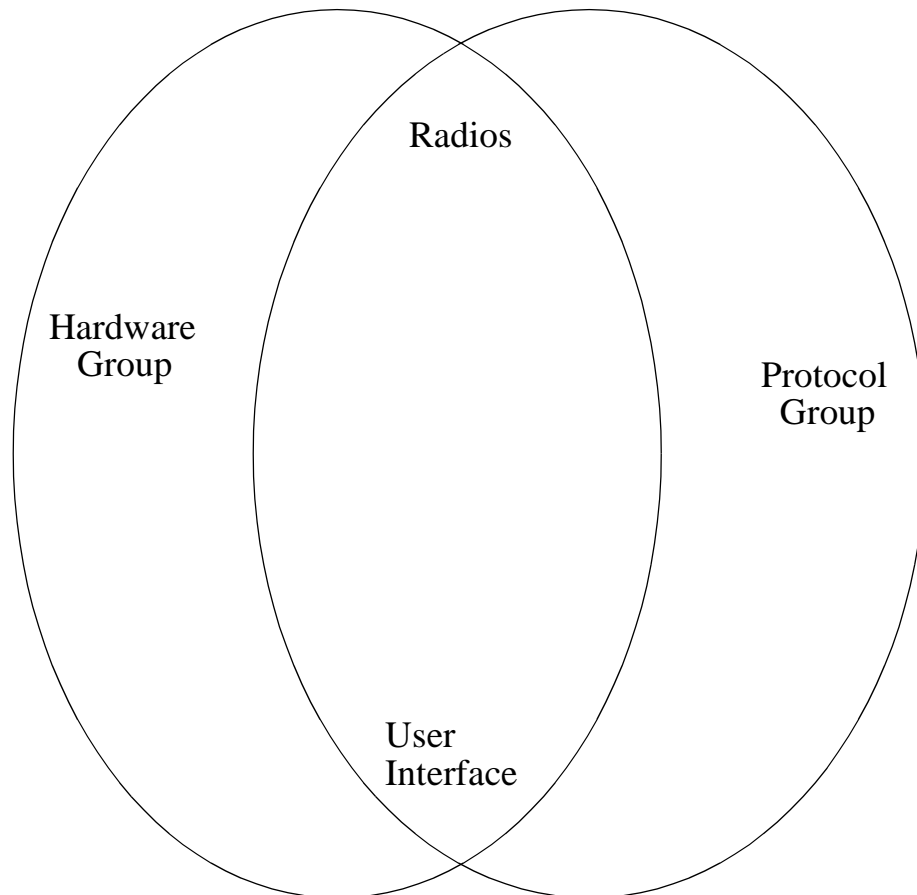


Figure 1: Intercom Project

There are several subdivisions within the Intercom Protocol group. Each subdivision represents an interrelated piece of a protocol stack: a series of dependent protocols from lowest level of hardware up to the highest level of control. Each protocol has its own functionalities, but depends on other protocols to properly relay information (refer to Figure 2 on page 6). Among the set of protocols to be designed and implemented is the control protocol, which is the focus of this paper.

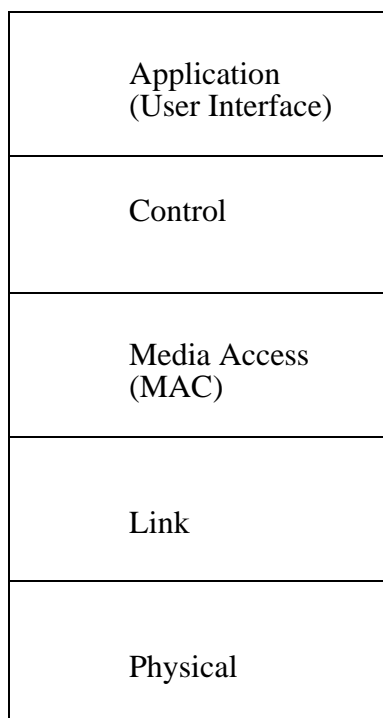


Figure 2: Protocol Stack

## 2 Control Protocol

The Control protocol is responsible for connection management between all objects (entities) of the system.

### 2.1 Methodology

Specification and description language (SDL) is a formal object-oriented language for communicating systems, originally designed for protocol specification. After in-depth review, it was concluded that it would be plausible to proceed with the design process using SDL. The Telelogic Tau tools available for this project implement SDL-oriented Object Modeling Technique (SOMT), which is a combination of SDL and a formal high-level design methodology

called OMT [4]. Telelogic tools are used to design abstract representations of a system. Telelogic has several inner tools which analyze, simulate and validate SDL models. A related investigation into the feasibility of the design process using SDL and Telelogic is detailed towards the end of this paper.

## **2.2 Research**

There was an ample amount of time spent reading and reviewing all of the necessary materials. During this phase, detailed notes were compiled on the areas of concentration to serve as a guideline. Since the control protocol will ultimately be represented as ‘C’ code targeted for an embedded processor, a preliminary investigation was conducted into the matter of the required amount of detail necessary for the model to generate accurate and usable code. Rough sketches were drawn to envision how abstract the SDL model of the control protocol could remain without crossing over into the level of implementation.

The information gathered from the preliminary investigation was used to begin the modeling phase. Once it was determined that enough material had been collected, it was time to focus on the control protocol itself.

## **2.3 Background/Specifications**

The functionalities for this particular system required that the control protocol have dynamic capabilities. It was also apparent that the control protocol needed to incorporate actions somewhat different from similar protocols. All information exchanged in the system takes the form of messages which are contained in a packet.

### **3 Design**

The design phase of the control protocol was fairly simple once all components had been defined. It was sensible to start at the highest, or most abstract, level of the control protocol and incorporate detail as the work progressed.

#### **3.1 Purpose**

The first area of focus for the control protocol was its purpose. Once it was established that the Intercom would support point-to-point and conference conversations for business purposes, it was easier to determine the control protocol's functionalities. The essential functionality was the message structure and meaning. Once the messages were defined, the signal paths and connection points could be established. It is very important that the messages are detailed to ensure all system functionality, yet flexible enough to allow for easy modification. At this stage, an abstract model of the control protocol is evident.

#### **3.2 Modeling**

The first step of the modeling phase was to take the information from the preliminary investigation and describe the general behavior. The goal was to start with an abstract model, and gradually add necessary information until an accurate system was formed. However, it was imperative to ensure, in this phase, that the system was not limited to a specific final form; there must be a bridge established for continued refinement, but the bridge must have missing links that will be added when the system is actually built. The abstract model consisted solely of signal paths and connection points (entities). Refer to Figure 3.

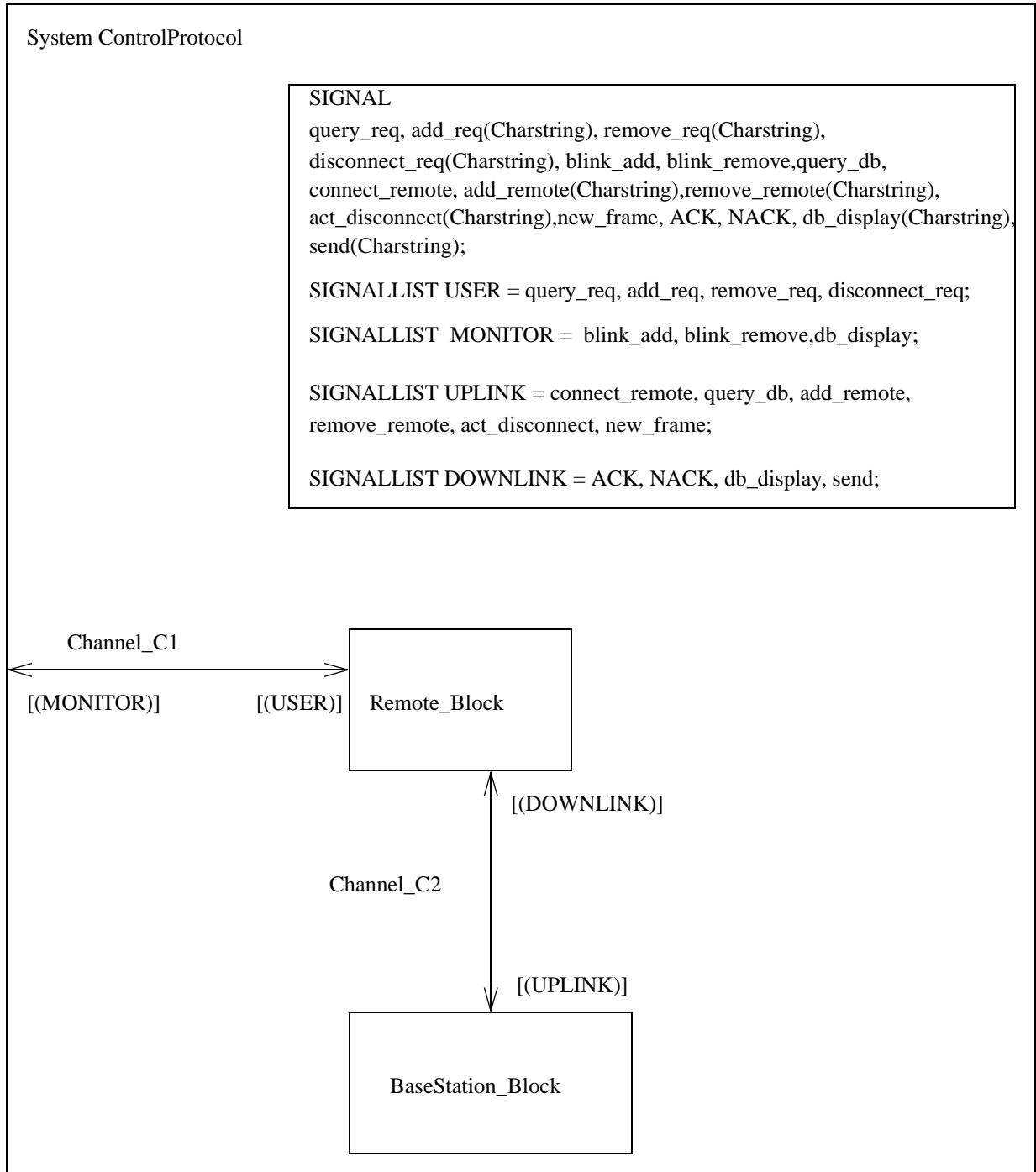


Figure 3: Abstract system model

From this model, the next logical step was to take one entity and further dissect its functionalities. For the purpose of clarity, the entity was evolved based on input and output signals. All steps required to describe the actions for all input signals were taken into account, and an abstraction was modeled. Then, each step was further broken down until enough functionality was evident to complete the desired action, but not so much that implementation flexibility is limited. This procedure was then repeated for all output signals. Once all actions had been modeled for the first entity, the same procedure was executed for the other entity. Once all actions for all entities were complete, the next step was to ensure that the proper links between entities had been established. At this stage, the modeling phase is considered to be complete. Now, the system model should incorporate all required signals, entities and actions (refer to Figure 4 on page 11).

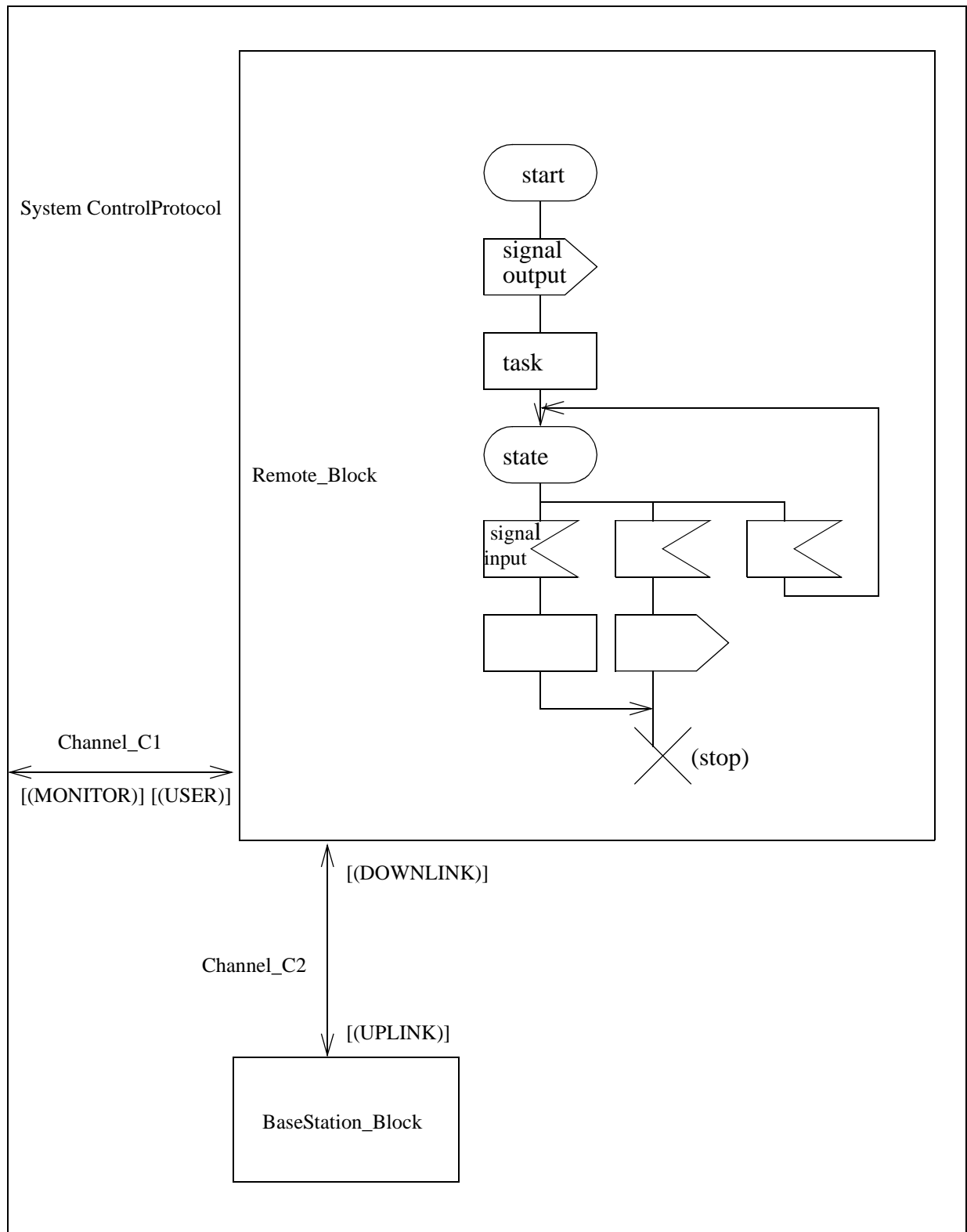


Figure 4: The system model showing remote block internal structure

## **4 Implementation**

The implementation phase consisted of analyzing the model and correcting all errors. After that was complete, Telelogic would be able to generate the C code, which was expected to be raw and in need of refinement.

### **4.1 Analyzing**

The debugging phase was the most tedious part of this design and implementation process. Since this model was written using the Telelogic Tau tool, the next step was to run the SDL analyzer provided by Telelogic. The analyzer first syntactically checks every statement in the model. If any syntax errors are found, they are corrected before proceeding. After the syntactical analysis was complete, the model was checked for semantic errors. The correction of semantic errors was rather involved. It often occurred that after correcting one semantic error, several syntax errors would appear. Throughout this process, it was evident that some of the errors, being syntactic or semantic, were present due to lack of knowledge about the correct format. This made it apparent that throughout this project, there was a need for continuous learning. Finally, the system model passed all syntactic and semantic errors. Once this was achieved, the last phase of the design and implementation process was entered.

### **4.2 Generating**

The generation of C code from the SDL model was the next and final phase. This procedure was relatively simple. Telelogic incorporates the ability to generate C code by converting all symbols and statements in the SDL language and model. All of this information is stored into a .c file of the same name as the system file. A makefile is also generated which contains the nec-

essary steps to effectively compile and link the newly generated C code. This was the final goal of the design and implementation of the control protocol for this particular research. The end result was over 50 pages of converted C code from SDL. The code will be reviewed and refined, and the radio will be built. For the author, this C code represents the completion of her portion of the project. However, this work will be continued by others.

## **5 Design Flow Feasibility**

The underlying research, as stated earlier, was to evaluate the design flow and determine the feasibility of the chosen language and tools.

### **5.1 SDL**

SDL appears to be a widely used language for various communications systems. However, based solely on this author's experience, SDL did not allow enough flexibility for the author to incorporate basic structures. For example, it was difficult to model the structure and usage of two-dimensional arrays. Given more time to learn SDL, the situation might have been different. However, due to the time constraints and the nature of the research, SDL was not the most efficient language.

### **5.2 Telelogic Tau**

Telelogic appears to be an extremely useful tool. However, while in the debugging phase, Telelogic was sensitive to errors that were not well represented. There are certain occasions when a programmer wishes to strangle the tool due to incorrect error representations, as was the case with this research. Since this is the case with most compilers, Telelogic cannot be harshly judged. Based on these facts, Telelogic was an efficient tool for this design flow.

## 6 Goals

At the start of this research, several goals were outlined. There were goals in regards to the technical aspects of the research. In addition, there were personal goals for the author, which were the main emphasis throughout this research. The goals are defined and detailed below.

### *1) Being an independent thinker.*

This goal is the first and most important. A researcher must have the ability to define a problem on their own and continue with problem solving independently. In the research environment, advisors try to be available for help but this is not always possible. An independent thinker does not need constant nurturing and assistance, and will speak up if there is a question or concern. Likewise, an independent thinker thrives on challenge and originality.

### *2) Understanding a real-world design methodology.*

Methodologies are well defined ways to go about solving a problem. The real-world design methodology used in this project (SDL) is based on object-oriented technologies. These technologies allow the designer to create a system abstraction which closely resembles system implementation.

### *3) Becoming familiar with the process of bringing an idea from conception to completion.*

This process involves continuous evaluation and refinement. It incorporates the need for continuous knowledge and research. Not one researcher can say they are an expert while involved in this process. The basis of this process is to constantly evaluate ideas and methods to determine which direction should be taken toward the end result.

#### *4) Learning the difference between design and implementation.*

Design incorporates modeling or creation. The design process helps to define the behavior and high-level structure of eventual code or schematics. The implementation is the actual writing of the code or schematics. The implementation stage should not be entered without a clear solution in mind.

#### *5) Finding out what it is like to be a part of a research/design project.*

Being a member of a research team requires a very involved role. There are no set hours for work; one works all the hours required to complete the project, There are few outside observers; all researchers have assignments which incorporate several sub-assignments. All work is valuable and essential to the overall project; the end results, for the most part, are tangible and evident.

## **7 Conclusions**

All information contained in this paper is based solely on this research and the author's experience with the Intercom Protocol group at the University of California at Berkeley. In conclusion, a control protocol was successfully created that will be an important part of the Intercom wireless system. Additional refinement of this protocol will be necessary, but the majority of the work has been completed. In addition, an investigation was conducted on the feasibility of the chosen design flow. It was determined that for this research, SDL was not the most efficient language and Telelogic was an efficient tool.

The design and implementation of any application involves several tedious steps. The design flow plays the most important role in accomplishing the goal of creating a specific tangible

end result. The investigation of this particular design flow will be used as part of the larger System Design research project.

## **8 Acknowledgments**

The author would like to take the opportunity to express gratitude to several key players in the evolution of this research. First and foremost, to the SUPERB program for the chance to participate. To all SUPERB staff who organized, coordinated, and lended support. Mainly, to mention by name, the SUPERB coordinators Sheila Humphreys and Janet Cooks. For the advisorship and trust given by Professor Jan Rabaey, who allowed the author to work with his research group. Most of all to Fred Burghardt, who served as an excellent mentor and team leader, without whom this research would have been a cumbersome sight. Lastly, not to overlook, the author's newly created "Cal Family" who realize that all work and no play makes for a dull researcher.

## **Resources**

1. J. Ellsberger, D. Hogrefe, and A. Sarma. SDL: Formal Object-oriented language for Communicating Systems. Prentice Hall: London, 1997.
2. R. Saracco, J.R.W. Smith, R. Reed. Telecommunications Systems Engineering using SDL. North-Holland: Amsterdam, 1989.
3. <http://infopad.EECS.Berkeley.EDU/intercom/> “The Intercom Project Home Page.”