This page intentionally left blank.

‘CALLS CONSIDERED HARMFUL’ AND OTHER OBSERVATIONS: A TUTORIAL ON TELEPHONY

Pamela Zave

AT&T Laboratories—Research

180 Park Avenue, Room B211
Florham Park, New Jersey 07932, USA
+1 973 360 8676
pamela@research.att.com

14 January 1997

‘Calls Considered Harmful’ and Other Observations: A Tutorial on Telephony

Pamela Zave
AT&T Laboratories—Research
pamela@research.att.com

Abstract. The software application domain of customer-oriented telephony is worth the attention of specialists in formal methods. Primarily this paper is a tutorial about customer-oriented telephony. It also includes observations about how this domain should be formalized, including critiques of some popular approaches.

1. Introduction

In the evolution of formal methods for software engineering, the time has come to develop formal methods for particular application domains [10,24]. This paper concerns the domain of customer-oriented telephony, i.e., software producing the externally observable behavior of voice telecommunications systems.

Customer-oriented telephony is worth the attention of industrial researchers and academicians alike, for the following three reasons:

Accessibility. Everybody uses telephones. Many characteristics of telephone systems are determined by international standards. Many consumer products now offer significant telephony functions. Anyone can study this domain, even without access to private intellectual property.

Importance. Telecommunications is widely predicted to be one of the key industries of the 21st Century. Both its political/economic context and its technological base are changing rapidly.

Trouble. Customer-oriented telephony software has all the usual problems of complex, long-lived, distributed, high-performance software systems. Several characteristic problems, such as feature interaction and the intertwining of separable concerns, are particularly visible and severe.

Customer-oriented telephony has already been the subject of much research activity, including invention of new specification languages and methods, implementation of new tools and environments, workshops, case studies, and other research projects. Nevertheless, there seems to be widespread ignorance of some of the basic principles and technological foundations of telephony. Naïveté mars much of the published work, and either compromises its usefulness or makes its usefulness difficult to evaluate.

This paper is a tutorial on customer-oriented telephony for the formal-methods community. It is intended to give someone who is interested in this software domain a good start, with enough depth and perspective to avoid egregious errors. Sections 2 through 5 delimit the domain and explain the relevant facts. Section 6 presents observations and conclusions about formal descriptions of this domain, including critiques of some popular approaches. It explains why the *call model*, which is the

foundation of most formal descriptions of telephony, is limited and potentially harmful.

Although there is plenty of tutorial material to be found in the networking literature, the presentation of telephony in this paper is unique. For one thing, its audience is different from the expected readership of a networking journal. For another thing, there is an unusual emphasis on relating different kinds of telephone system: how they are similar, how they are different, and how they interact. This emphasis is necessary to convey the crucial information in a small space, but it also has the important advantage of revealing the inherent coherence of the application domain. As a result of the coherence newly revealed here, it seems possible that all kinds of telephone system can be specified with the same techniques.

2. Boundaries of the Domain

Figure 1 shows the world-wide telephone network decomposed into two (highly distributed) machines, one performing voice transmission and one performing customer-oriented telephony. We are interested in the behavior of the upper machine, which is implemented exclusively in software.

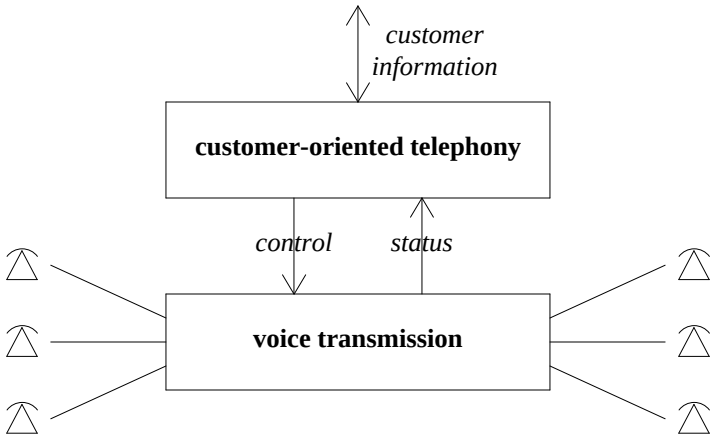

Figure 1. A decomposition of the world-wide telephone network.

The relationship between the two machines is similar to the relationship between two adjacent layers in the OSI Reference Model.[1] The customer-oriented machine sends control commands to the voice-transmission machine. The voice-transmission machine sends status information to the customer-oriented machine, including command results and notifications of stimuli from telephones. The *call-*

[1]The split between the two machines shown in Figure 1 appears to fall within the application layer of the OSI Reference Model [19].

processing function of the customer-oriented machine is to emit control commands at this interface.

The customer-oriented machine also has other interfaces and functions, all concerning customer information. It must accept information about customer identity, service preferences, payments, etc. It must emit information about bills, service usage patterns, etc.

There is plenty of software in the voice-transmission machine, so the decomposition shown in Figure 1 does not separate software from hardware. Rather, it separates software functions directly observable by customers from software functions for resource management: hardware monitoring, hardware fault diagnosis, resource allocation (including network routing and the creation of voice paths), performance tuning, and interaction with operators who participate in the resource-management functions.

Telecommunications is concerned with transmission of data and multimedia as well as plain voice (telephony); often all of these media are transmitted on the same physical network. Why separate telephony from the rest of telecommunications?

From the perspective of users, data transmission is fundamentally different from telephony. Multimedia communication, on the other hand, is an extension of telephony. I recommend attacking the problems of telephony first because they are extremely difficult in themselves, yet have a certain familiarity and coherence that may enhance intuition. Better to solve these first and then extend the results to multimedia (which appears possible [18]) than to attempt everything at once and get nowhere.

3. Overview of Voice Transmission

This section is an overview of the voice-transmission machine. The information in it is necessary for two reasons: (1) the call-processing function of the customer-oriented telephony machine consists of controlling the voice-transmission machine, and (2) the capabilities of the voice-transmission machine limit the services that the customer-oriented machine can offer to its customers. The voice-transmission machine is also the least familiar part of telephony. The customer-information interface, in contrast, can easily be imagined by any computer scientist.

3.1. Components

Figure 2 shows some of the components of the world-wide voice-transmission machine (or "network," since it is highly distributed).

A *telephony device* is an input/output device for voice.[2] It might be a conventional telephone, speakerphone, cordless telephone, mobile telephone, personal computer equipped with a speaker and microphone, fax machine, answering machine, or many other things. The key characteristic of a telephony device is that it supports a single two-way voice channel. This makes sense because sound usually

[2]There is no industry-standard term for a telephony device. *Customer-premises equipment* comes close in meaning, but also includes PBXs (see Section 4).

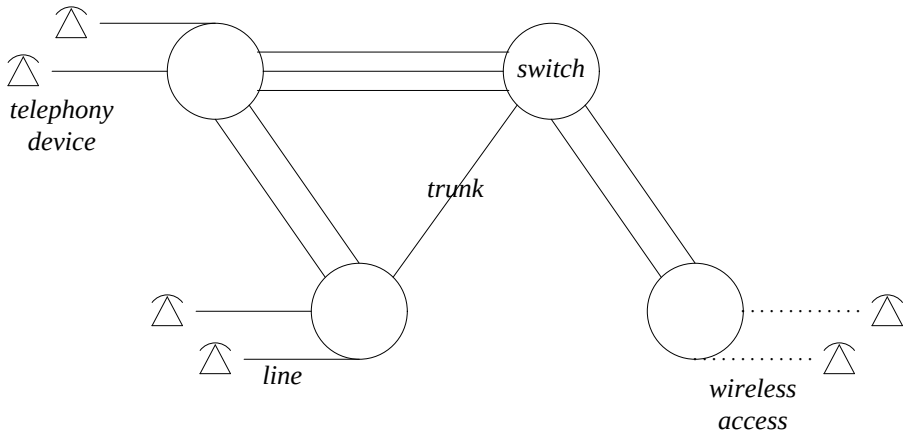

Figure 2. Some components of the world-wide voice-transmission network.

passes between the device and a human head through the air, and there would be no way to maintain the acoustical isolation of two voice channels.

A telephony device is usually connected to the network by a *line*, dedicated to the device and also capable of supporting a single two-way voice channel. The exception is a mobile telephone, which has wireless access. It is connected to the network (whenever it is connected) by a two-way radio channel allocated to the telephone only for the duration of the access episode.

When multiple (extension) telephony devices are plugged into the same line, the rest of the network perceives their collective actions as the actions of one device. Since the telephone systems in the network cannot distinguish the presence or absence of extensions, specifications can ignore their existence completely.

A line leads from a telephony device to a *switch*, which is a voice-handling node of the network. Switches are connected by many *trunks*, each of which is like a line in supporting a single two-way voice channel. A switch is capable of connecting any two trunks or lines coming into the switch, for the purpose of creating a voice path (see Section 3.3).

The components shown in Figure 2 are the ones that support voice transmission directly. In other words, Figure 2 is a picture of a circuit-switched network. In addition to the circuit-switched network, there is a separate packet-switched *signaling network* used for transmitting control messages among the switches.

In addition to signaling among switches, there is also a need for signaling between telephony devices and switches. If the device's line is digital, then it has a two-way signaling channel separate from its voice channel. If the device's line is analog, then the voice channel is also used for signaling (see Section 3.2).

3.2. Voice Processing

Many switches contain (or are closely associated with) special hardware devices for voice processing. For example, a *three-way conference bridge* is a device that connects three two-way voice channels. It mixes its three voice inputs so that each of its three voice outputs is the normalized sum of the other two inputs. Conference bridges can be built to mix almost any number of voice channels. The important thing about conference bridges is that a more-than-two-way conversation is never possible without one.

Another example of voice processing would be recording and playback of speech. By far the most common use of voice processing, however, is for *in-band signaling*, in which control signals are transmitted on the voice channel. In-band signaling is defined in contrast to *out-of-band signaling*, in which control signals are transmitted through a separate signaling channel or network.[3]

In-band signals to a telephony device take the form of tones or announcements (a voice-processing device generates the tones and either plays recorded announcements or synthesizes speech from textual announcements). In-band signals from a telephony device usually take the form of touch-tones, which are versatile and easily detected by voice-processing hardware. Keywords and sound/silence transitions can also be recognized as control signals.

3.3. Voice Paths

A *voice path* allows persistent, two-way voice transmission between two endpoints.[4] An endpoint is usually a telephony device or a voice-processing device within a switch. An endpoint might also be just a loose end within a switch, in which case the other endpoint is *on hold.* A voice path can pass through any number of switches and trunks.

Many telephony features manipulate voice paths, as illustrated by the sequence from (a) to (d) in Figure 3. In (a) there is a voice path between the left and right telephones. In (b) the left telephone has put the right telephone on hold and obtained a path to a middle telephone; there are now two voice paths. In (c) the three telephones are conferenced, each having its own path to a conference bridge, so that there are now three voice paths. In (d) the left telephone has ordered a transfer, dropping out of the conversation while leaving the other two telephones joined by a single voice path.

Let us consider how a path such as the one shown in Figure 3(a) could be set up and torn down. The protocol used to bring each line or trunk into the path could be different, but all the protocols are similar. Figure 4(a) is a "message sequence chart" illustrating one kind of setup. The control signals between any two nodes refer to the line or trunk that is being added to the path, and travel on signaling channels associated with the line or trunk (which might be the same as the voice channel of the line or trunk, in the case of in-band signaling).

Figure 4(a) shows only the two most important types of control signal,

---

[3] Voice processing is usually called *signal processing.* I have used a different term to distinguish voice processing in general from its particular use for in-band signaling.

[4] A voice path is a *circuit* in the terminology of circuit-switched networks.

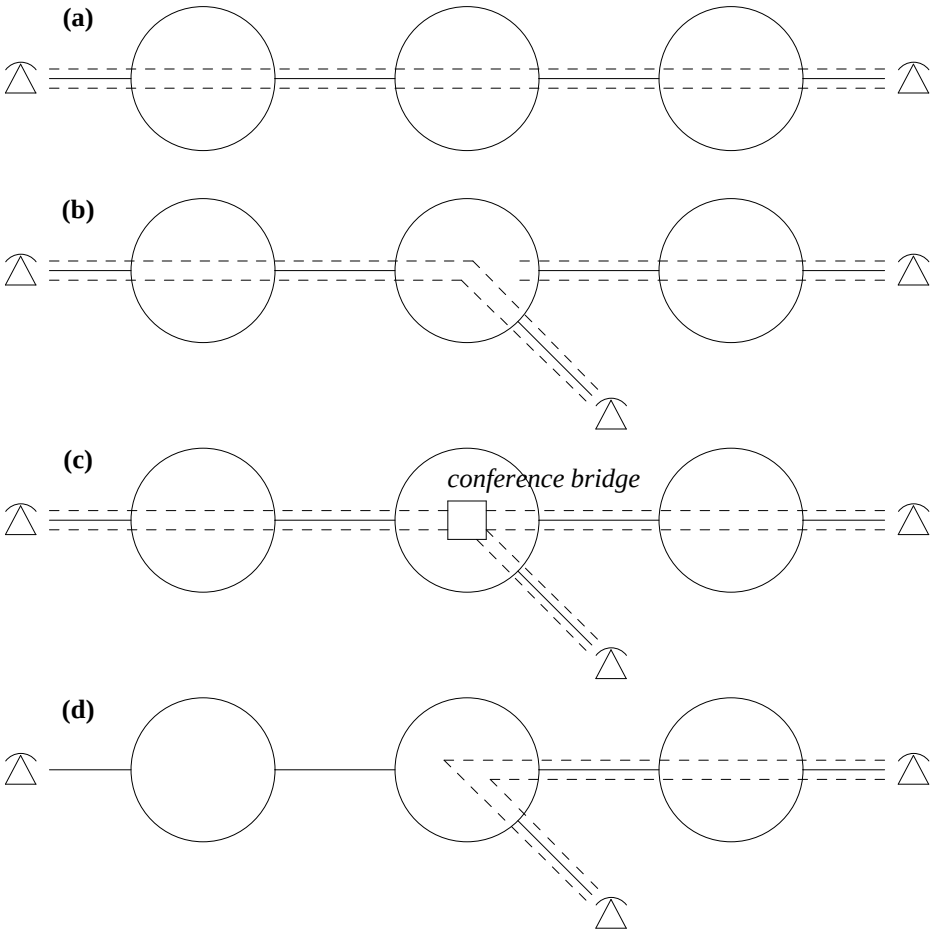

Figure 3. Some examples of voice paths.

generically named *request-path* and *complete-path*, although the protocol between any two particular nodes might use other signals as well. The path is set up when the last *complete-path* signal reaches the left telephone.

When each switch in Figure 4(a) receives a *request-path* signal, there are decisions to make. At the level of the voice-transmission machine, there is always a physical routing decision. At the level of the customer-oriented telephony machine, there may be a decision to redirect the path (perhaps because the current destination has requested call forwarding), or to treat the request specially in some other way.

The path can also be set up in stages, as shown in Figure 4(b). Here switch *S2* takes a very active role. It first completes setup of a path from the left telephone to

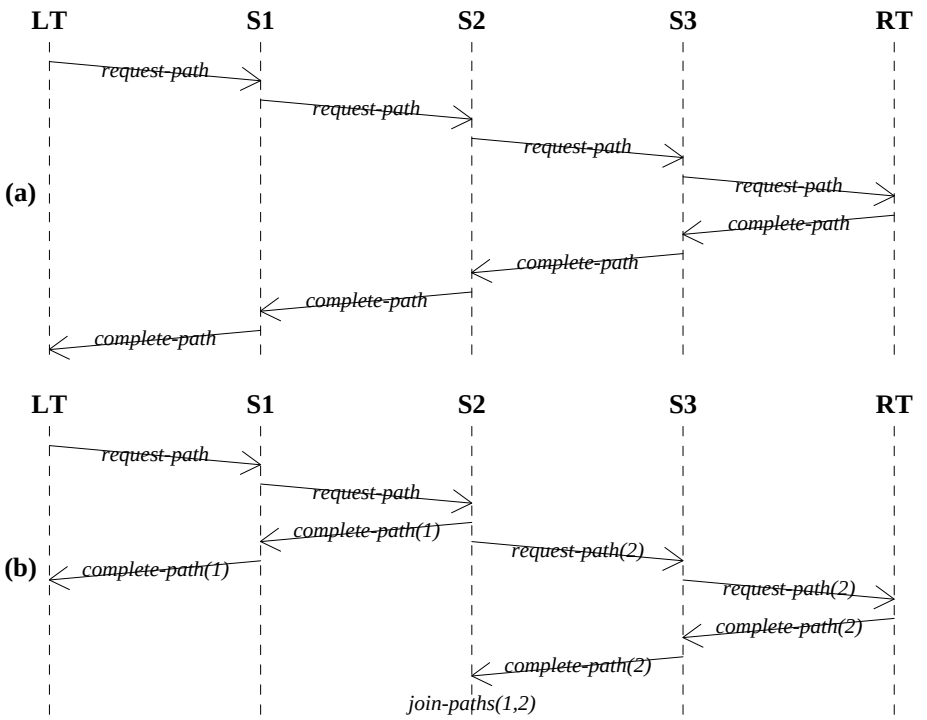

Figure 4. Two scenarios for setting up a voice path.

itself, then initiates setup of another path from itself to the right telephone. When the second path is complete it joins the two paths, creating a single voice path between the left and right telephones.

Why set up a voice path in multiple stages? There are many possible reasons, but the most common one is the need for in-band signaling. Suppose that, when the path request reaches *S2*, the customer-oriented machine needs more information about the request than comes with the request signal. Further suppose that the protocol or protocols active between *S2* and the left telephone have no built-in features for obtaining and transmitting the additional information. Then the only remaining option is to complete the voice path between *S2* and the telephone, so that *S2* can solicit and collect the additional information through in-band signaling. An example of this situation will be given in Section 4.

Many other protocol details can influence a specification of telephony services, depending on its level of abstraction. Consider, for example, how a busy tone is produced. In one common method, the switch directly connected to the destination (busy) telephone completes setup of the voice path and then connects its end of the path to its own busy-tone generator. The advantage of this method is that the busy tone heard at the originating telephone corresponds to the geographic region of the

destination telephone. In another common method, a negative acknowledgment of the path request is sent back to the switch directly connected to the originating telephone. Upon receiving this signal, the originating switch completes the voice path to the originating telephone (if it has not already done so) and connects the path to its own busy-tone generator. The advantage of this method is that it minimizes use of voice-transmission resources.

Roughly speaking, providers of telephone service bill for completed voice paths (and sometimes feature usage as well). But there are many details and exceptions to be considered. In the two busy-tone methods above, there is never a bill, even though there is always some completed voice path. In the case of a completely successful path request, the path is usually completed at the time that the destination telephone starts to ring, to ensure that a voice path will be available when the telephone is answered. Billing does not usually begin, however, until the destination telephone is actually answered.

The teardown of a voice path is similar to its setup. Either end can initiate the teardown, and the other end must acknowledge it. A path can be torn down all at once, in which case the generic *release-path* and *release-ack* signals might have the same general pattern as shown in Figure 4(a). It might also be torn down in stages, in which case the signals might exhibit the same general pattern as Figure 4(b).

4. System Boundaries Within the Domain

Figure 5 is another picture of the world-wide voice-transmission network. Its main difference from Figure 2 is that the boundaries of *telephone systems*, owned and operated by different *service providers*, are also shown.

A *private branch exchange (PBX)* is a private switch, usually found on the premises of a business or institution. A *local exchange carrier (LEC)* provides local service; it may run a local network or simply a single switch. An *interexchange carrier (IEC)* provides a long-distance network. A *national system* provides telephone service for an entire country, combining the functions of LEC and IEC systems. A *cellular system* provides mobile service; it may reach the rest of the world through any other type of system.[5,6]

Needless to say, any particular software-development project is going to be confined within the boundaries of a single telephone system. We are now in a position to understand the similarities and differences among the various types of telephone system.

One difference is that most telephone systems are distributed, while some (e.g.,

---

[5] Other combinations are possible. For example, in the United States, there will soon be systems that act like national systems in the sense of combining the functions of LEC and IEC systems, and that act like IEC systems in the sense of having direct competition and needing the cooperation of other service providers for access to some local telephones.

[6] An Internet-based telephone system is harder to include in this diagram, both because the Internet is an overlay network and because it is a data network. When a user of an Internet telephone service wishes to speak to someone who does not subscribe to the same Internet telephone service, then his call goes to the public network through a *tail-end hop-off* feature. Tail-end hop-off is exactly like the interface between a LEC system and an IEC system.

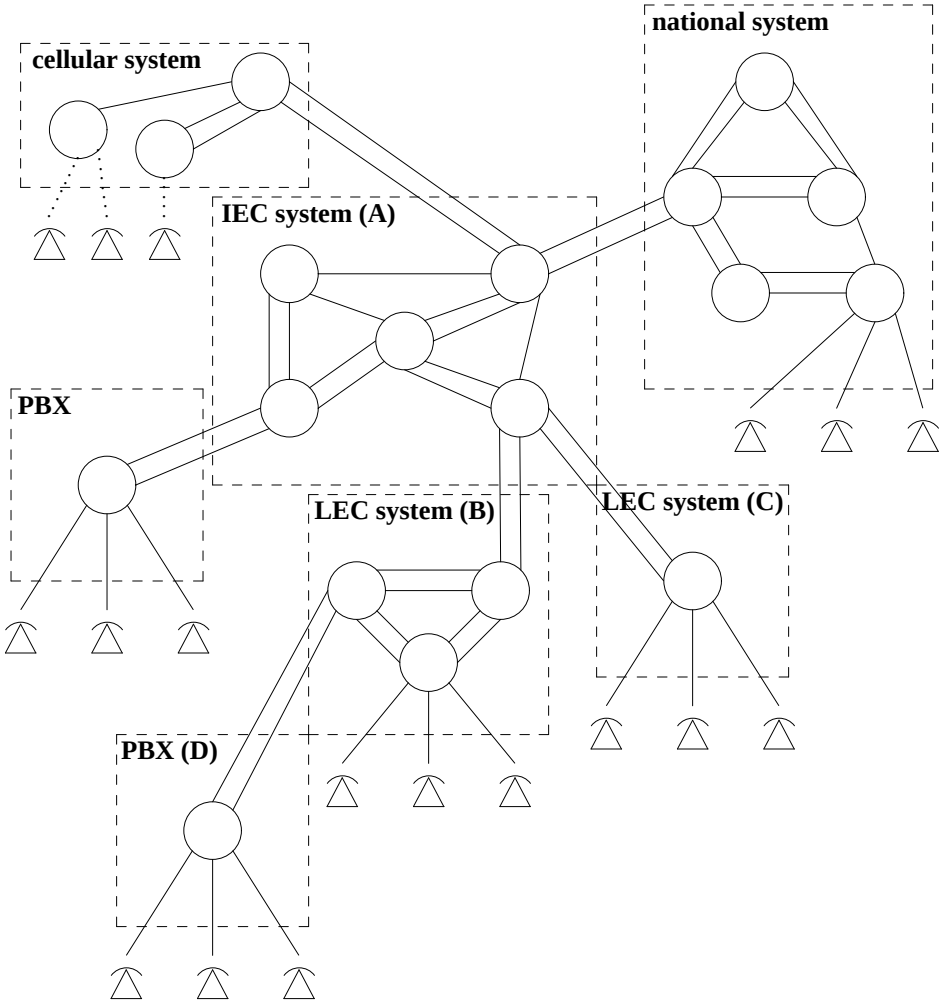

Figure 5. Some systems and components of the world-wide voice-transmission network.

PBXs) are not. This difference is discussed in Section 6.1.

The most important distinction among telephone systems is that some systems have direct access to telephony devices, while others do not. There is often a special, knowledge- and feature-rich relationship between a telephone system and a direct-access device. The system knows what kind of device it is, and whether it is busy or idle. If the device is complex (e.g., has many buttons) then the protocol used on the line to that device probably has similar complexity (many signals). In contrast, a system that interacts with a telephony device through another system usually has

minimal knowledge of the remote device, and can only know about the device's state what another system tells it.

Although the difference between direct and indirect access is significant, it should not be overstated. *No* telephone system is free of indirect access, because no telephone system can reach all telephones by itself. A typical LEC system has direct access to many telephones, but the LEC system (e.g., *B* in Figure 5) has the same indirect relationship to telephones connected to a local PBX (e.g., *D*) as an IEC system (e.g., *A*) has to the LEC system's direct-access telephones. Furthermore, providers of all types of telephone system now aspire to offer roughly the same capabilities, including routing, screening, billing, multiplexing, conferencing, and messaging features.

When a voice path includes an intersystem trunk, there is cooperation between the two adjacent systems to set up the path. At present this is essentially the only means of interoperation. Interoperation can only be made more powerful by enriching the current protocols between systems, and by providing economic incentives for service providers to support the enriched protocols.

In the absence of richer interoperation, all telephony features beyond *plain old telephone service (POTS)* are implemented strictly within individual telephone systems. Here are two examples.

If an IEC system offers a credit-card feature, then setup of voice paths using that feature probably looks something like Figure 4(b), where switch *S1* is in the LEC system serving telephone *LT*, and switch *S2* is in the IEC system. As explained in Section 3.3, when the path request reaches the IEC system at *S2*, the IEC system discovers that it is a request for credit (probably because of the dialed string, which is part of the *request-path* signal), and needs to collect a credit account number. It completes the voice path to *LT*, prompts for and collects the number through in-band signaling, and then (if the account number is good) extends the path to *RT*. Meanwhile the LEC system serving *LT* has nothing to do with the credit-card feature, and need not even detect that it is being used.

Also consider a conference among three telephones accessed directly by systems *B*, *C*, and *D* respectively. *The conference feature could be provided by any one of the systems A, B, C, or D.* The bridge is located in the system providing the conference feature, and the other systems see nothing more than plain voice paths.

In a typical conference, one telephone plays the role of the controller, and is the only device with the power to add parties, drop parties, or transfer (in Figure 3 the left telephone is the controller). Thus the controlling telephone must have a way of transmitting conference-control signals to the system providing the conference.

If the controlling telephone and the conference-providing system have a direct-access relationship, then the solution to this problem centers on their line protocol, which must include signals for conference control. For example, if it is an analog line, then an existing signal such as a *flash* will acquire a special meaning in the conference context (this is how the *three-way calling* feature works [5]).

If the controlling telephone and the conference-providing system have an indirect-access relationship, on the other hand, then in-band signaling is the only possibility, because intersystem protocols do not support conference control. In this case a voice-processing device will be attached, by the conferencing system, to the

path from the controlling telephone. Because the device will be attached in *monitor mode*, it will monitor the voice input from the controlling telephone for touch-tones or keywords, without interrupting the voice path in any way.

5. The Call-Processing Interface

This section concerns the interface that the customer-oriented telephony machine uses to control call processing in the voice-transmission machine, as shown in Figure 1.

When we are concerned with a single telephone system, we are dealing with a vertical slice of Figure 1, as shown in Figure 6. The customer-oriented telephony machine lies within the system and is isolated from other telephone systems. The lateral interfaces of the voice-transmission machine are the lines and external trunks by which it is connected to the rest of the world-wide network, including both their voice channels and associated signaling channels (if any).

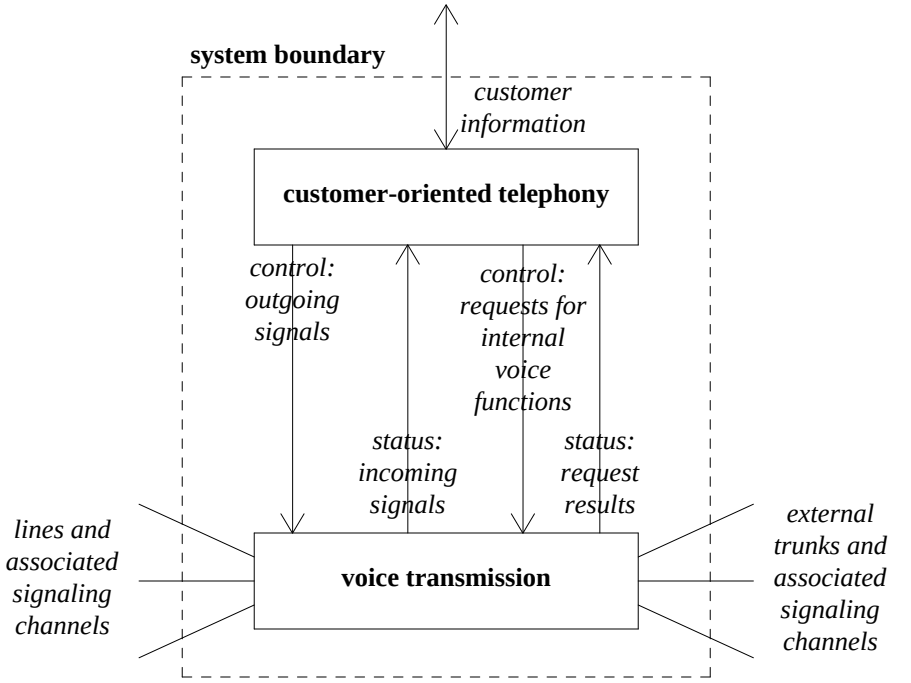

Figure 6. The call-processing interface within a telephone system.

The voice-transmission machine reports to the customer-oriented machine signals received from lines and external trunks. Some of the commands from the customer-oriented machine to the voice-transmission machine instruct the latter to send out control signals on the lines and external trunks. The other instructions from

the customer-oriented machine are requests for voice functions internal to the system.

As you might expect, the perspective needed for the customer-oriented layer is quite different from the natural perspective of the voice-transmission layer. Looking inward from the lines and external trunks, the primary concern of the customer-oriented machine is to bring about *connections* among sets of lines and external trunks. A connection is a relationship whose presence enables mutual voice communication. The primary concern of the voice-transmission machine, as we have seen, is voice paths. Voice paths are different from connections, for the following two reasons.

(1) Connections are always local to one telephone system. Making connections is the main thing that a telephone system does for its customers, and the system must have some control over all the ingredients used to make them. Voice paths, in contrast, are global. They can pass to or through telephony devices and external switches, neither of which is under the system's control.

(2) Connections describe telephony at a higher level of abstraction than voice paths do. For example, at the level of the customer-oriented machine, a three-way conference is simply a ternary connection relation. At the level of the voice-transmission machine, it is implemented using a conference bridge and the local sections of three voice paths. For another example, a voice path leading from a telephony device to a voice-processing device within a system might not be involved in any connection made by that system. It might be implementing a user interface to one telephone (see Section 6.2) rather than a communication relationship among telephones.

Although Figure 6 gives a rough idea of the information that must pass across the call-processing interface, it leaves many choices open: what the level of abstraction is, how the functions are grouped, how the information is filtered, how much of the architecture of the voice-transmission machine is revealed, etc.

A clean call-processing interface is desirable for two reasons. One reason, which is widely acknowledged in the telecommunications industry, is the need to upgrade the voice-transmission machine easily without incurring the delay and expense of changing the customer-oriented machine. As everyone knows, transmission technology is improving rapidly.

To elaborate this point, in the traditional architecture of a telephone network, all the network nodes have dual functions. They are components of the voice-transmission machine, in which role they serve as switches. They are also components of the distributed customer-telephony machine, in which role they provide feature logic and store customer data. The current industry trend is toward the Intelligent Network architecture (see also Section 6.3), in which the two layers in Figure 6 are implemented on completely disjoint sets of network nodes. In the Intelligent Network architecture, the switches are concerned only with voice transmission and related functions. The customer-oriented telephony machine is implemented completely within other network nodes specializing in feature logic and customer data.

The other reason for a clean call-processing interface, which is less widely understood, is pure separation of concerns. Each of the layers is subject to many pressures toward change, growth, and increased complexity. When the two layers

intertwine, an increase in the complexity of one tends to increase the complexity of the other, enabling a feedback loop with very unfortunate consequences.

6. Observations

6.1. On Distribution

Most specifications of telephony features assume that the voice-transmission machine is centralized without justifying the assumption or even mentioning it (e.g., [20]). Some specifications deal honestly with the fact that it is not, at great cost to their writers and readers (e.g., [17]).

The stark reality is that the features of real telephone systems are extremely complex. At present we cannot specify them completely even with the centralization assumption, let alone without it.

It makes sense to use the centralization assumption for feature specification, even when the voice-transmission machine is distributed and the assumption is false. Techniques such as those suggested by Jacob [13] might make it possible to specify a distributed system as if it were centralized, while maintaining a suitable formal relationship between the specification and the thing specified. Even without the formal relationship, a specification based on this assumption is much better than no specification at all. Telephone engineers are very well accustomed to working with imperfect specifications, and implementing them to "reasonable and customary" standards.

Use of the centralization assumption has another valuable consequence: telephone systems in which the voice-transmission machine is centralized and in which it is distributed can both be specified using the same techniques.

6.2. On User Interfaces

The "user interface" of telephony consists of what a person using a telephony device hears, sees (on the device's displays), and can do to choose or affect services. A telephone system implements a particular user interface by emitting and accepting certain signals associated with lines and external trunks. This section consists of three simple observations about these interfaces.

(1) Every telephone system has to provide more than one user interface. Systems with direct-access interfaces must also provide indirect-access interfaces. Systems with only indirect-access interfaces always have external trunks with different protocols. Wireless access has some characteristics of indirect access and some of direct access; the cellular systems that provide wireless access must also provide indirect access. Despite this variety of interfaces, I have never yet seen a paper on telephony specification even mention multiple interfaces, let alone provide them.

(2) In specifying user interfaces, it is extremely important to unify in-band and out-of-band signaling. Both are heavily used, and they are used for overlapping functions. Despite this obvious fact, I know of only one paper that even mentions in-band signaling (it includes a proposal for unifying the two kinds of signaling in interface specifications [27]).

(3) The literature on protocol specification may seem relevant, but the protocol approach is not necessarily the best approach to telephony.

In the protocol literature, there is a "constraint-oriented style" of specification in which the specification is decomposed into local (endpoint) and end-to-end constraints [3,21,23]. This style has been used for telephony specification in LOTOS [8]. In the LOTOS specification, the local constraint is a description of the user interface for a line. The end-to-end constraint is a description of the relationship between two interfaces brought about by the telephone system for the purpose of connecting their lines.

An important fact about the constraint-oriented style is that the local constraints are pure projections of the end-to-end constraints [3]. In other words, the local constraints can be derived from the end-to-end constraints. This makes sense for protocol specification, where end-to-end constraints are arguably the only important properties, but it does not make sense for telephony because of the following telephone capability.

Some telephone systems provide user interfaces for other purposes than for establishing connections. Interactive voice interfaces enable customers to update databases and retrieve information. Interactive voice interfaces are also used to collect information such as authorization codes, directory numbers, and credit account numbers as part of screening, routing, and billing features, respectively. Note that these latter interfaces are active before any attempt to connect is made.

Thus the behavior of a user interface is often independent of any connection attempt. By allowing it to stand on its own, we increase separation of concerns and decrease redundancy. While it is true that the implementation of some interfaces involves voice paths that extend into a telephone system to reach voice-processing devices deep within its network, as explained in Section 5, that is a detail with no place in a specification.

When interfaces are treated realistically, the specifier often faces the complexity of a many-to-many mapping between interface and connection events. Some specification techniques have been developed to manage this complexity [28].

6.3. Calls Considered Harmful

A *call* is an attempt by one telephone (the *caller*) to establish one connection to one other telephone (the *callee*). The state of the call encodes or implies information about the states of all three entities. A *call model* is a conceptual model for telephony that describes all telephony in terms of calls.

The most advanced call model in use today is the Intelligent Network Conceptual Model (INCM). It was developed under the auspices of the International Telecommunication Union (ITU) and the European Telecommunication Standard Institute (ETSI), and is being promulgated as a standard by those organizations [7,9,12,15]. The primary purpose of the INCM is to support evolution toward the Intelligent Network architecture, as introduced in Section 5.

The INCM defines points in a call where control of the call can pass from a switch to a feature node and back again. A few control point names and definitions [15] show the nature of the INCM:

Address Collected: This point identifies that the destination address has been received from the user.

Busy: This point identifies that the call is designated to a user who is currently busy.

Active State: This point identifies that the call is active and the connection between the calling and called parties is established.

Clearly these control points are combined states of a caller interface, a connection or potential connection, and a callee interface or potential callee.

Some features fit into the call model very well. Here are some examples of features and capabilities that the INCM is designed to handle.

Translation. Translation from one directory number to another is used for forwarding, 800/freephone calling, and other services. It can occur at any point before an attempt to connect is made.

Screening. Screening features are used to authorize and deny calls. Like translation features, they can be invoked at any point before a connection attempt is made, and have no effect on later stages of the call.

Queueing. Some callers compete for the attention of a pool of service agents. Calls are queued, and connected to available agents in FIFO order. Queueing can be represented as a nondeterministic preconnection delay in the history of a call.

Many other features and telephone capabilities, however, subvert the one-to-one-to-one correspondence that is the essence of the call model. Up to a point, the call model can be extended (patched) to accommodate these features. Beyond that point, the call model is likely to break down in hopeless complexity. The following examples present some of the features and capabilities that most deeply undermine the assumptions of the call model.

Conferencing. Conferences connect sets of interfaces, from three to a hundred. The call model, of course, is concerned with only two interfaces. For a large conference, an appointment must be made ahead of time. Thus, at its starting time, a large conference can be thought of as a connection with no interfaces. Furthermore, the conference can even initiate creation of the voice paths to all the participating telephones—very different from a call, which is always initiated by a caller.

Serial and time-multiplexed connections. A credit-card customer enters account information, then proceeds to make a series of calls on the same account before hanging up (he presses "#" on the dialpad to end a call without hanging up). In this situation, one interface makes a series of independent connections. Alternatively, customers with call waiting or multibutton telephones can time-multiplex several independent connections from within the same interface.

Delayed communication. A voice message is like a call in the sense that there is a caller and a callee, and the callee hears what the caller says. But the caller and callee are never connected to each other, and are not even accessing the system at the same time! Furthermore, the telephone system offering the messaging service might call or be called by the recipient of the message. Automatic-callback features also introduce multiple communication phases, separated by time.

Interface-only features. As discussed in Section 6.2(3), many telephone features provide user interfaces without or before any attempt to make a connection.

Indirect access. If we take seriously the idea that a call connects telephones

(rather than lines or trunks), then a long-distance call is a multi-system phenomenon, and an IEC system (providing only indirect access) can never complete a call on its own.

These counterexamples suggest to me that the call model is as much of a hindrance as a help in understanding telephony. It is easy enough to see how its dominance arose. Partly it is the overwhelming historical influence of POTS. Partly it is the presence of an important point-to-point concept in telephony—but the concept is the voice path, which belongs at the physical level and not at the logical level. And partly it is the immediate practical problem that we have nothing better to put in its place.

6.4. How Formal Methods Can Help: Feature Interaction

Telephone engineers do not need formal methods to help them implement POTS: there have been successful implementations of POTS for about 75 years. These final two sections describe aspects of telephony in which help is needed, and for which the help needed seems to be the kind that formal methods can provide.

Most generally, the feature-interaction problem [5,11,25] is the problem of making a telephone system behave the way we want it to despite continual, incremental expansion of services. This has proved to be a very difficult problem; despite much attention from researchers it has scarcely been alleviated.

There is a range of formal approaches to the feature-interaction problem. At one end of the range is the detection approach, in which features are specified independently in a compositional language. The composed features are analyzed algorithmically to detect inconsistencies and failure to satisfy desired properties (e.g., [1,2,4,16,20]).

At the other end of the range is the structural approach, emphasizing modular specification and separation of concerns. The idea is to organize the specification so that desirable properties are guaranteed by its structure, and so that it is easy to add features without destroying its structure or exploding its complexity. The notion that features can or should be specified independently receives less emphasis than in the detection approach (e.g.,[6,14,26,29]).

These two approaches have complementary advantages and disadvantages. The following two comparisons capture the most important points.

(1) Detection research is more straightforward to carry out. A researcher can simply apply an existing language and analysis tool to the problem and see what happens. In-depth knowledge of telephony is not usually needed and seldom influences the results.

Structural research, on the other hand, is groping in the darkness. Researchers cannot assume that existing languages and tools are adequate. The more knowledge of telephony available, the better. As a result of all these factors, structural research often leans heavily on the structure of the implementation, thus compromising the call-processing interface.

(2) In a pungent critique of detection research [22], Velthuijsen observes that no one has yet succeeded in using algorithmic analysis to detect a major feature interaction that was not previously known. The reason is that features are almost

never orthogonal—in almost all cases, adding a feature creates exceptions and requires changes to the previously specified features. The goal property checked by the algorithm must incorporate all of the exceptional cases. By the time a person has written the property correctly, he already understands all of the exceptions and potential interactions, both desirable and undesirable.

Even if the detection approach succeeds to perfection, how can the specification errors be corrected so as to produce a well-structured, readable specification of the whole system? My experience suggests that the corrections will form a cascade of ugly and unmanageable exceptions.

The structural approach does not suffer from these disadvantages. Rather, its whole focus is to avoid them by eliminating exceptions and providing a readable overall specification.

These two comparisons show that there is no clear winner. Nor does there need to be, since the two approaches can be combined (in fact, many of the examples cited mix elements of both). Nevertheless, I believe that the structural approach is more fundamental and more necessary than the detection approach. The structural approach is the one that seeks to discover and exploit knowledge of the application domain.

There is one other possible approach that deserves some attention. It would be very helpful to have a robust collection of simple, abstract properties of well-behaved telephone systems ("principles of telephone etiquette"). For example, consider the principle, "A subscriber is never billed for a call unless he knows he is paying for it." This principle would be violated by a system that sets up collect calls without informing the callee. This principle might also be violated through a call to an 800/freephone number, if the 800 number translates to a normal 900 number.[7]

Such principles are difficult to find, as Velthuijsen's remarks make clear. Success would require working at a much more abstract level than the properties used for detection, formalizing vague concepts such as "what a subscriber knows he is paying for." Success would also require giving the principles prescriptive, as well as descriptive, force, since some current features are sure to violate them.

Such principles would support the other approaches in obvious ways. For one example, they could be checked by detection mechanisms. For another example, they could be used to constrain or derive the detailed behavior of features within the boundaries of a structural approach. But the hierarchy of abstract telephony concepts needed to construct the principles would provide a great deal of insight and organization in their own right.

6.5. How Formal Methods Can Help: Separation of Concerns

The clean call-processing interface described in Section 5 is more of a goal than a reality. There is widespread eagerness to find a specific interface that will separate the concerns of these two layers effectively while preserving maximum flexibility on both sides.

---

[7] In addition to the transmission cost of the call, a caller to a 900 number also incurs a per-minute charge payable to the callee.

Acknowledgments

I am indebted to Michael Jackson for many years of fruitful discussions about telephony. Anthony Finkelstein, Daniel Jackson, Gerald Karam, Nils Klarlund, and Jim Woodcock also provided helpful comments.

References

[1] Johan Blom, Roland Bol, and Lars Kempe. Automatic detection of feature interactions in temporal logic. In K. E. Cheng and T. Ohta, eds., *Feature Interactions in Telecommunications Systems III*, pages 1-19. IOS Press, 1995.

[2] Johan Blom, Bengt Jonsson, and Lars Kempe. Using temporal logic for modular specification of telephone services. In L. G. Bouma and H. Velthuijsen, eds., *Feature Interactions in Telecommunications Systems*, pages 197-216. IOS Press, 1994.

[3] Gregor V. Bochmann. A general transition model for protocols and communication services. *IEEE Transactions on Communications* XXVIII(4):643-650, April 1980.

[4] Kenneth H. Braithwaite and Joanne M. Atlee. Towards automated detection of feature interactions. In L. G. Bouma and H. Velthuijsen, eds., *Feature Interactions in Telecommunications Systems*, pages 36-57. IOS Press, 1994.

[5] E. Jane Cameron, Nancy D. Griffeth, Yow-Jian Lin, Margaret E. Nilson, William K. Schnure, and Hugo Velthuijsen. A feature interaction benchmark for IN and beyond. In L. G. Bouma and H. Velthuijsen, eds., *Feature Interactions in Telecommunications Systems*, pages 1-23. IOS Press, 1994.

[6] D. Cattrall, G. Howard, D. Jordan, and S. Buj. An interaction-avoiding call processing model. In K. E. Cheng and T. Ohta, eds., *Feature Interactions in Telecommunications Systems III*, pages 85-96. IOS Press, 1995.

[7] José M. Duran and John Visser. International standards for intelligent networks. *IEEE Communications* XXX(2):34-42, February 1992.

[8] Mohammed Faci, Luigi Logrippo, and Bernard Stepien. Formal specification of telephone systems in LOTOS: The constraint-oriented style approach. *Computer Networks and ISDN Systems* XXI:53-67, 1991.

[9] James J. Garrahan, Peter A. Russo, Kenichi Kitami, and Roberto Kung. Intelligent Network overview. *IEEE Communications* XXXI(3):30-36, March 1993.

[10] J. A. Goguen and Luqi. Formal methods and social context in software development. In *Proceedings of the Sixth International Conference on Theory and Practice of Software Development (TAPSOFT '95)*, pages 62-81. Springer Verlag LNCS 915, 1995.

[11] Nancy D. Griffeth and Yow-Jian Lin. Extending telecommunications systems: The feature-interaction problem. *IEEE Computer* XXVI(8):14-18, August 1993.

[12] ITU-T/ETSI Recommendations Q1201-Q1205, Q1211, Q1213, Q1214. 1993.

[13] Jeremy L. Jacob. Refinement of shared systems. In John McDermid, editor, *The Theory and Practice of Refinement: Approaches to the Formal*

Development of Large-Scale Software Systems, pages 27-36. Butterworths, 1989.

[14] Yoshiaki Kakuda, Akihiro Inoue, Hiroyuki Asada, Tohru Kikuno, and Tadashi Ohta. A dynamic resolution method for feature interactions and its evaluation. In K. E. Cheng and T. Ohta, eds., *Feature Interactions in Telecommunications Systems III*, pages 97-114. IOS Press, 1995.

[15] Jalel Kamoun. Formal specification and feature interaction detection in the Intelligent Network. Department of Computer Science, University of Ottawa, Ottawa, Ontario, 1996.

[16] Yasuro Kawarasaki and Tadashi Ohta. A new proposal for feature interaction detection and elimination. In K. E. Cheng and T. Ohta, eds., *Feature Interactions in Telecommunications Systems III*, pages 127-139. IOS Press, 1995.

[17] Andrew Kay and Joy N. Reed. A rely and guarantee method for Timed CSP: A specification and design of a telephone exchange. *IEEE Transactions on Software Engineering* XIX(6):625-639, June 1993.

[18] Evan H. Magill, Simon Tsang, and Bryce Kelly. The feature interaction problem in networked multimedia services: Past, present and future. EPSRC No. GR/K 72995, October 1996, `http://www.comms.eee.strath.ac.uk/~fi/fimna.html`.

[19] N. Mitra and S. D. Usiskin. Relationship of the Signaling System No. 7 protocol architecture to the OSI Reference Model. *IEEE Network* V(1):26-37, January 1991.

[20] Tadashi Ohta and Yoshio Harada. Classification, detection, and resolution of service interactions in telecommunication services. In L. G. Bouma and H. Velthuijsen, eds., *Feature Interactions in Telecommunications Systems*, pages 60-72. IOS Press, 1994.

[21] K. J. Turner and M. van Sinderen. LOTOS specification style for OSI. in Jeroen van de Lagemaat and Tommaso Bolognesi, editors, *The LOTOSPHERE Project*, pages 137-159. Kluwer Academic Publishers, 1995.

[22] Hugo Velthuijsen. Issues of non-monotonicity in feature-interaction detection. In K. E. Cheng and T. Ohta, eds., *Feature Interactions in Telecommunications Systems III*, pages 31-42. IOS Press, 1995.

[23] Chris A. Vissers, Giuseppe Scollo, Marten van Sinderen, and Ed Brinksma. Specification styles in distributed systems design and verification. *Theoretical Computer Science* LXXXIX(1):179-206, 1991.

[24] Pamela Zave. Application of formal methods is research, not development. *IEEE Computer* XXIX(4):26-27, April 1996.

[25] Pamela Zave. Feature interactions and formal specifications in telecommunications. *IEEE Computer* XXVI(8):20-30, August 1993.

[26] Pamela Zave. Secrets of call forwarding: A specification case study. In *Formal Description Techniques VIII (Proceedings of the Eighth International IFIP Conference on Formal Description Techniques for Distributed Systems and Communications Protocols)*, pages 153-168. Chapman & Hall, 1996.

[27] Pamela Zave and Michael Jackson. Requirements for telecommunications services: An attack on complexity. In *Proceedings of the Third IEEE*

International Symposium on Requirements Engineering, pages 106-117. IEEE Computer Society Press, 1997.

[28] Pamela Zave and Michael Jackson. Where do operations come from? A multiparadigm specification technique. *IEEE Transactions on Software Engineering* XXII(7):508-528, July 1996.

[29] Israel Zibman, Carl Woolf, Peter O'Reilly, Larry Strickland, David Willis, and John Visser. Minimizing feature interactions: An architecture and processing model approach. In K. E. Cheng and T. Ohta, eds., *Feature Interactions in Telecommunications Systems III*, pages 65-83. IOS Press, 1995.