

# H.325: An Application Platform

## A Closer Look at the “Container”

Paul E. Jones  
Rapporteur Q12/16  
April 7, 2009



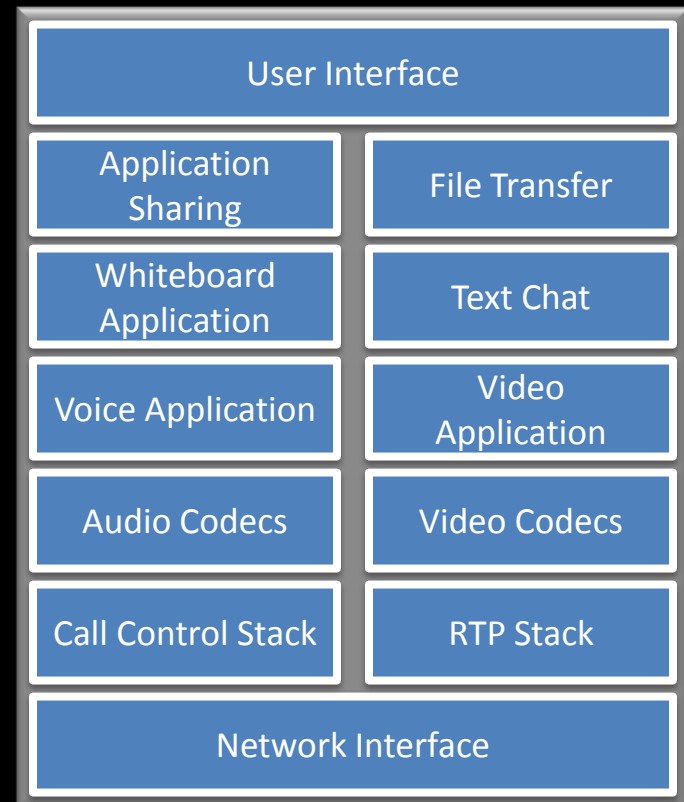
# Significant Differentiation

- What significantly differentiates H.325 from legacy systems is that “Voice over IP” or video conferencing are viewed as “**applications**” and anybody could define any kind of new application to integrate with other applications
- **Numerous applications** may be created for H.325, including voice, video, app sharing, and a flashing lamp to tell you an incoming communication has arrived
- H.325 is an application platform wherein the user may **utilize and number of applications** to communicate using whatever means is most appropriate or natural

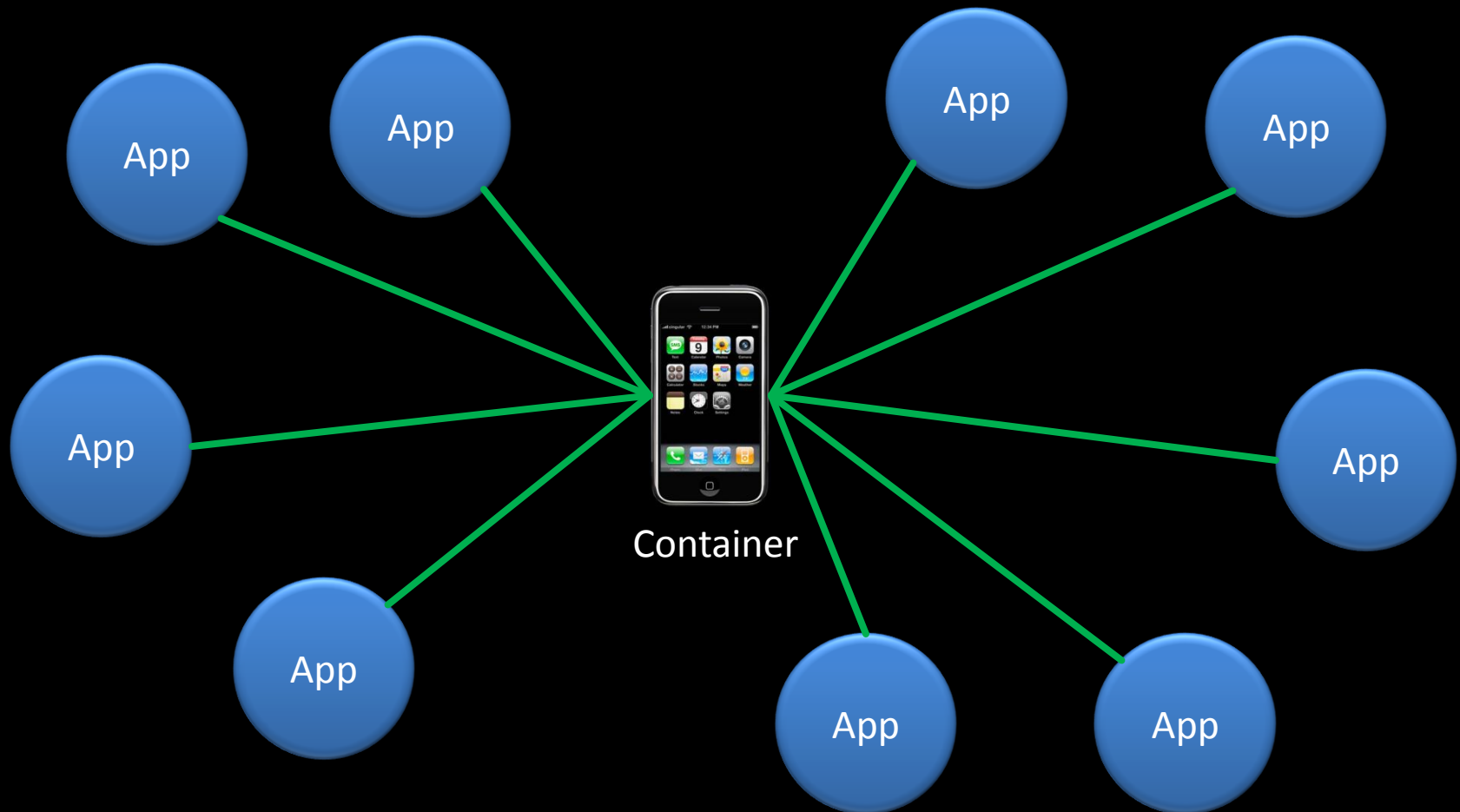
# The Legacy Systems

- All functionality is in one fat, **monolithic** application
- **Development teams cannot work independently** or release functionality independently
- **No means to enable third-party developers** to introduce new application functionality
- **New applications require upgrades** to the endpoint
- **Upgrades** are “all or nothing”; upgrading thousands of users **requires significant planning in development phase!**
- **Legacy systems do not** take full advantage of the **distributed** nature of the Internet
- The net result is that most **legacy** systems are only “**voice over IP**” systems

## User Agent



# Container Enables Many Distributed Applications across Many Devices



# The “Container”

- The “Container” is the user’s **control device** and identity in an H.325 environment
- Container might be any kind of device, but will most **likely a mobile device**



# Where Applications Reside

- In the Container
- On a PC
- In a TV panel on the wall
- In a desk phone
- In a whiteboard in the room
- In a lamp sitting on a table
- In the refrigerator
- In the network
- Or **any place** else!

# Applications...

- Communicate **locally**
  - Event notification (e.g., to enabling the flashing lamp)
  - To allow music to stream from one device to another
  - To copy a file from one device to another
- Communicate **remotely**
  - To enable voice or video communication
  - To enable file transfer the remote person or entity
  - To enable play video games over the Net
- Communicate **both** locally and remotely
  - Allow movement of audio from a mobile to a desk phone
  - Allow movement of video from the PC to an LCD panel

# Further, Applications...

- **Are not burdened** with locating other users or applications
- **Are not burdened** with end-to-end signaling messages
- **Focus** only on what the application needs to do
- Are free to perform any function the application developer can **imagine**
- May be **implemented in any language**
- May exist **on any physical device**
- Are **independently developed** from the Container or other network entities
- Are **intelligent** and understand what they need to do; they are **not controlled by a central server** as in the legacy softswitch model

# Simplicity through Decomposition

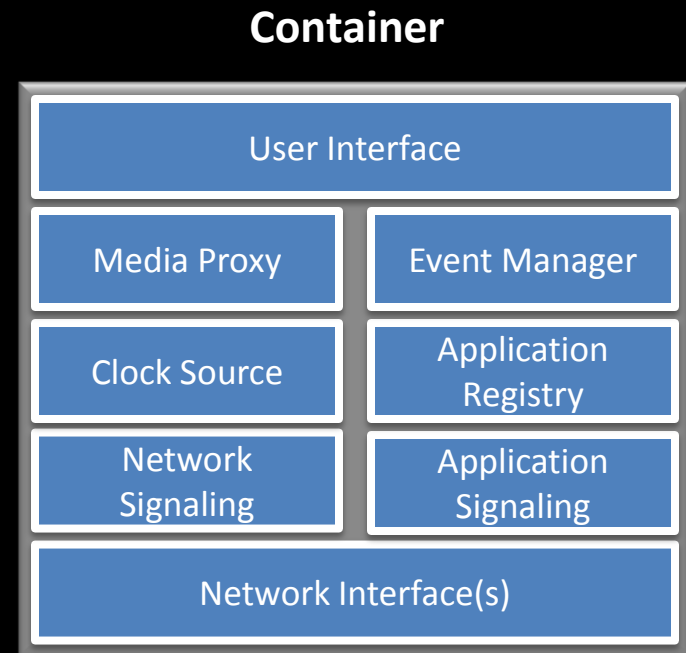
- Any communication system must necessarily have **some complexity**
- H.325 strives to make each element as **simple as possible**
- H.325 will **not sacrifice functionality** in applications
- Different **development teams can work independently** on different parts of the system, including the Container, the various applications, and the network functions

# Building a Reliable System

H.325 will be designed to  
**work** properly, **negotiate**  
**capabilities**, and be a **robust**  
communication system

# Structure of the Container

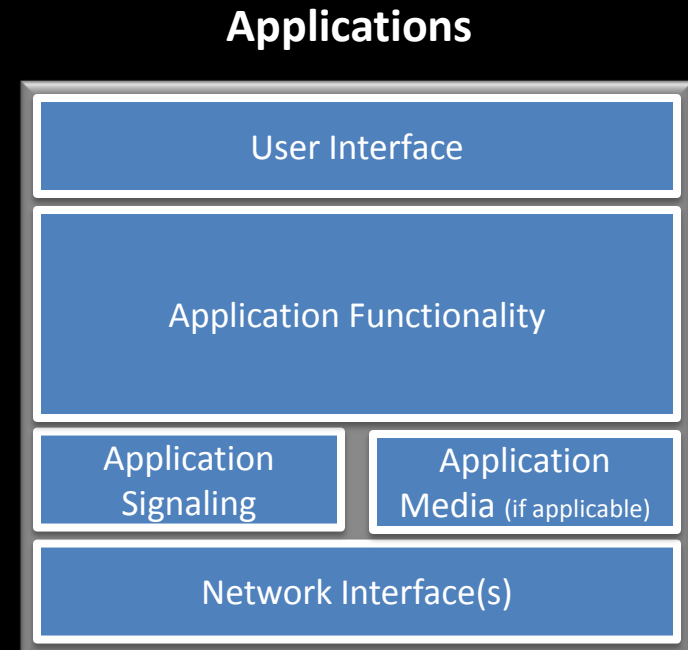
- The **Container** is responsible for providing a **user interface**, network signaling, and coordinating communication between applications
- The **Container** is **not responsible for** actually implementing or even understanding **application behavior**
- The Container is *not* a “voice over IP” system
- The Container is *not* the device, but just a functional element



Note: These functions are still being defined and this list is only to illustrate the separation of “container” logic and “application” logic

# H.325 Applications

- Applications **may exist on the same** device as the Container **or separate devices**
- **H.325 does not impose restrictions** on application functionality; developers are free to be creative
- A user may **employ any number** of applications at the same time
- Applications may **communicate** with other applications, either **locally or remotely**
- Applications may be **upgraded independently** of the Container or other applications
- The **ITU will standardize some applications** to ensure interoperability
- **Non-standard applications** will offer value to users, as developers can be creative and introduce a host of applications that **improve productivity**



Note: Beyond communicating with the Container and processing certain required messages, most of the application is focused on application functionality that is entirely separate from other entities in the system

# Cloud-Based Applications

- Since the application **user interface is “remotable”** (i.e., one can interact with applications from other applications or the Container), applications could be implemented as Web-based services that run in the Internet
- One could employ **part of the application in the cloud** and part of the application as a locally installed device or application
- The **“web browser” is an Application** for H.325 and could be invoked by another application (e.g., to enable a person who calls into an audio bridge to automatically join a data conferencing bridge)

# Simplifying Applications Running on a Personal Computer

- An application should only focus on what the application does
- We might want to develop an “**application stub**” that runs on a computer to allow applications to **interact** with a Container **more easily**
- This further **reduces application complexity**, by moving all common behavior to a single place
- This is a significant architectural consideration that will **enable quick adoption of new wireless technologies** that do not require changes to applications
- The Container would still interact with each application individually, as if each application contained all of the stub logic and application logic; there is **no difference externally**

## Personal Computer

